UNIVERSIDAD CARLOS III DE MADRID
Departamento de Ingeniería Telemática

PROYECTO FIN DE CARRERA

Retos para la banca en la era digital. Nuevo modelo y arquitectura de servicio de las oficinas.

Autor: **Alberto Corisco Nieto**

Tutor: **Julio Villena Román**

Leganés, Septiembre de 2015

Título: Retos para la banca en la era digital. Nuevo modelo y arquitectura de servicio de las oficinas.

Autor: Alberto Corisco Nieto

Director: Julio Villena Román

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

A mis padres, sobre todo a ellos.

A Angi por su incansable comprensión y apoyo estos meses.

A la zanahoria y a aquél que la sostiene pues esa motivación ha supuesto el impulso definitivo.

A todos los que directa o indirectamente, de forma consciente o no, han contribuido a que llegue este día.

"Si no lo llevas dentro no lo puedes sacar soplando"
Louis Armstrong, trompetista.

"¡Así que sopla! ¡Sopla bien fuerte y que se vea!"
Yo.

Resumen

En los últimos años, las entidades del sector de la banca minorista están viendo como surgen nuevos competidores, provenientes en su mayoría del sector tecnológico. La rápida digitalización del sector y un cambio en las expectativas por parte de los clientes, está obligando a las empresas tradicionales de banca a reaccionar para recuperar el terreno perdido. Se exigen profundos cambios y nuevos enfoques. Lo que no es una tarea sencilla teniendo en cuenta que la cultura imperante en estas entidades es poco proclive al cambio.

En este escenario digital y ubicuo, la razón de ser de las oficinas, tal cual se concibe actualmente, ha quedado algo desdibujada. Sin embargo, una de las herramientas más poderosas y diferenciadoras de que disponen las entidades de banca es, precisamente, la red de oficinas. Éstas aportan el valor intangible de la cercanía y el trato personal. Utilizadas de la forma adecuada puede ser el factor diferenciador que decante la balanza en su favor. Pero para que el cambio sea perceptible habrá que emprender una tarea de profunda autocrítica, redefinir la filosofía y objetivos de la empresa.

Una vez hecho esto, el primer paso para poder proponer métodos, productos y servicios más actuales y atractivos pasa, obligatoriamente, por renovar la base tecnológica que sustenta las operaciones diarias de clientes y empleados. Para poder plantar batalla a los nuevos competidores digitales, esta infraestructura debe ser moderna, dinámica y flexible.

El presente documento pretende dar respuesta a parte de esas cuestiones y plantear una arquitectura tecnológica que permita a una entidad acometer los cambios requeridos.

Palabras clave: oficina, banca minorista, arquitectura de servicios, computación en la nube

Índice general

Índice de ilustraciones

Índice de tablas

Capítulo 1

Introducción y objetivos

1.1 Motivación del proyecto

La llegada de Internet, hace ya años, cambió la forma en que trabajamos, nos relacionamos y entendemos el mundo. Dicha revolución ha impactado en todos los sectores en mayor o menor medida. El sector de la banca, conservador por naturaleza, lleva años adoptando las nuevas tecnologías tanto internamente como proyectándolas hacia el cliente como factor diferenciador. Sin embargo, su modelo de negocio, y la forma en que lo aborda, apenas ha cambiado más allá de reflejarse en Internet.

Muchas empresas del sector tecnológico han sido conscientes de ésta situación y la han aprovechado en su favor explotando nuevos productos, creando nichos de mercado y renovando los medios de interacción con el cliente. Actualmente, las actividades de estos competidores abarcan desde medios de pago hasta correduría de bolsa, pasando por transacciones peer-to-peer, tarjetas de crédito, envíos de efectivo, financiación y préstamos. Todo ello apoyado en su base de conocimiento tecnológico. Este compendio de actividades y la forma en que las llevan a cabo, es una de las amenazas más importantes del sector bancario en su conjunto.

En muchas entidades financieras existe la certeza de que el correcto uso de la tecnología puede suponer un factor diferenciador determinante respecto a la competencia. La digitalización del modelo de negocio (su presencia en Internet) es algo ya extendido y

real sin lo que una entidad actual no puede pretender subsistir. Superado este hito, se ha de plantar cara a las nuevas amenazas. Para ello es necesario plantear nuevas estrategias de producto y formas de interacción con el cliente, que supongan mejoras en imagen, satisfacción y captación de clientes, así como un ahorro de costes. Para que sea fructífera, toda esta evolución debe ser rápida e ir apoyada en avances tecnológicos.

Históricamente, las entidades bancarias, especialmente las españolas, han hecho énfasis en tener una mayor presencia y accesibilidad de cara al cliente. Esto se ha conseguido mediante la creación de una gran cobertura de oficinas. Sin embargo, la transformación en banca digital, ha desdibujado el objetivo de dicha red de oficinas. A día de hoy, el sector casi al completo tiene el foco puesto en el negocio digital y su necesidad de contrarrestar las amenazas provenientes del sector tecnológico. Por tanto, se augura una drástica reducción en el número de oficinas existentes. Sin embargo, es innegable que pese al alto coste de su mantenimiento, el valor intangible de la cercanía y el trato personal que se obtienen en una oficina difícilmente puede conseguirse por otros medios.

La solución radica en abordar la modernización de la banca desde un punto de vista holístico en el que se tenga en cuenta todos y cada uno de los factores que concurren en la actual situación. En este contexto, las oficinas tendrán que evolucionar para sobrevivir. Se deben explorar, por tanto, formas innovadoras de recuperar la importancia del canal presencial como medio de interacción con el cliente. El presente proyecto se engloba justamente en este conjunto de iniciativas.

1.2 Objetivos

Como ya se ha expuesto previamente, es necesaria una modernización general en el modelo de negocio de las entidades bancarias y, más particularmente, de su red de oficinas. Pero los retos a superar no provienen sólo de agentes externos, existen otros factores internos como pueden ser plataformas tecnológicas anticuadas, cultura interna inmovilista, poca predisposición al cambio y falta de conocimiento en las nuevas Tecnologías de la Información y Comunicaciones (TIC). Todo ello ha contribuido a la vigente situación del sector.

En cuanto a oficinas se refiere, el modelo existente se basa en unos principios desfasados que no encajan con las necesidades ni expectativas actuales de los clientes, altamente influenciados por las nuevas TIC. Esta deriva en el modelo de negocio y el funcionamiento de las oficinas ha propiciado un escenario en el que es difícil distinguir entre oficinas de distintas entidades. Además, el catálogo de productos y servicios ofrecidos apenas ha cambiado en años y el papel sigue siendo el principal medio documental. Asimismo, la experiencia del cliente no es rica ni continua entre los distintos canales.

El objetivo principal de este proyecto es, por tanto, el de analizar y plantear un rediseño del concepto de oficina bancaria, actualizándolo al contexto actual. Para que esta renovación tenga éxito, debe apoyarse en la potencia y versatilidad de las TIC actuales y adoptar procesos y enfoques que los clientes utilizan diariamente en otros aspectos de su vida.

En primer lugar, será necesario analizar, tanto formal como técnicamente, la situación actual de las oficinas. Una vez encontradas las debilidades concretas, habrá que plantear un nuevo modelo de oficina que las solvente. Dicha solución deberá formularse bajo las premisas de mejora en la interacción, captación y satisfacción de cliente y avance en términos de eficiencias y costes. Para lograrlo, se hará especial énfasis en los siguientes puntos:

- Redefinir la forma de aproximación e interacción con el cliente.
- La diferenciación como constante en el diseño.
- Modernizar la imagen de la oficina.
- Acentuar las estrategias de ventas de la oficina bancaria.
- Primar la experiencia del cliente.
- Integración en las estrategias de omnicanalidad.

Además, se deberá proponer una arquitectura global y de oficina que sustente el nuevo modelo. Esta infraestructura tecnológica habrá de diseñarse para que soporte las operaciones actuales bajo el modelo propuesto y disponga los medios necesarios para la implantación de un renovado catálogo de productos y servicios. En este apartado se tendrán en consideración varios puntos:

- Ahorro de costes frente al modelo actual.
- Reducción en el consumo de ancho de banda de los procesos.
- Tiempo de disponibilidad de la plataforma.
- Seguridad.

Con tales fines, se recurrirá al uso de arquitecturas de servicios y elementos de computación en la nube. Ello permitirá componer un ecosistema en el desplegar nuevos servicios de forma ágil y con una alta calidad de servicio.

1.3 Estructura de la memoria

El presente documento se estructura en seis capítulos. El primero de ellos, éste, expone una breve introducción al proyecto. En él se realiza un acercamiento al problema que se tratará a lo largo del proyecto y se exponen los objetivos que éste pretende satisfacer.

En el segundo capítulo se expone con más detalle el problema al que pretende dar respuesta este proyecto. Se realiza un análisis general de la actual situación del sector bancario y, más particularmente, de la red de oficinas de las entidades.

El tercer capítulo está dedicado a plantear, desde un punto de vista formal, un nuevo modelo de oficina, que solucione los problemas anteriormente identificados, y los elementos que la componen.

El capítulo cuatro recoge, de forma más técnica y analítica, el diseño de una arquitectura TIC que de soporte el nuevo paradigma de oficina planteado en el capítulo previo. Para ello se exponen y analizan lo que serán los dos pilares básicos de la solución: la arquitectura de servicios y la computación en la nube. Posteriormente, se procede a una descripción del diseño tecnológico de cada una de las piezas que componen el modelo de oficina propuesto.

El siguiente capítulo, el quinto, expone el desarrollo de un caso de uso concreto en la arquitectura propuesta.

En el sexto, y último, capítulo se exponen las conclusiones del proyecto y el trabajo futuro.

Capítulo 2

La oficina como punto de acceso del cliente

El proyecto actual surge de una motivación de cambio, de una necesidad de diferenciación y modernización del modelo de oficina existente. Como paso previo al diseño de la solución es necesario hacer una breve introducción al escenario actual del sector en su conjunto y, más profundamente, al de las oficinas bancarias: su planteamiento, funcionamiento y la plataforma tecnológica sobre la que se soportan los procesos de negocio.

2.1 Retos para la banca en la era digital

En los últimos años han surgido una serie de amenazas para los integrantes del sector bancario. Estas amenazas provienen indistintamente de start-ups, empresas de la era digital ya consolidadas, fabricantes de software o hardware, cadenas de venta minorista, operadores de comunicaciones, etc. Aparentemente, un grupo tan heterogéneo no parece tener ningún punto en común que justifique incluirlos en un grupo que pone en riesgo la posición del sector financiero. Sin embargo, analizando en detalle las recientes iniciativas de estas organizaciones se puede apreciar que todas ellas comparten dos características importantes. Por un lado, han sabido ver una demanda subyacente de nuevos tipos de

productos financieros y formas de consumirlos. Por otro, han basado su estrategia de venta y relación con el cliente en el uso intensivo de las TIC.

Una encuesta conducida por Accenture a 40.000 norteamericanos en 2014 [ACC1] arroja conclusiones muy interesantes, a la par que alarmantes, sobre el estado del sector de banca minorista:

- Más de un 25% de clientes estaría predispuesto a operar con bancos que no tuviesen red de oficinas.
- Cerca de un 75% de clientes considera que su relación con su banco es meramente transaccional.
- Aproximadamente la mitad de los encuestados consume o estaría dispuesto a consumir productos financieros de empresas de fuera del sector como pueden ser Paypal, T-Mobile, Costco, Apple, Amazon, Walmart, Canada Post, eBay, etc.

Estas conclusiones se refieren a todo el espectro de encuestados en su conjunto. No obstante, si se desgranan los resultados por edad, las cifras se acrecientan de forma llamativa. Las nuevas generaciones viven la tecnología de forma más intensa y tienen menor apego a las empresas y modelos establecidos. Indiscutiblemente, pues, existe una clara tendencia alcista respecto al consumo y aceptación de este tipo de servicios.

El abanico de servicios ofrecidos por los competidores digitales abarca desde cuentas corrientes (Walmart, Bancorp, Moven, Simple), pagos y transacciones peer-to-peer (Paypal, Dwolla, Square, Amazon Payments, Google Wallet), tarjetas de crédito (Google Wallet, Walmart, T-Mobile), envíos de efectivo (Walmart, MoneyGram), financiación y préstamos (Costco, Prosper, OnDeck), correduría de bolsa (Robinhood, Scottrade), etc. La situación no es muy halagüeña si tenemos en cuenta que gran parte de los ingresos de la banca minorista provienen de este tipo de actividades (p. ej. los ingresos provenientes de pagos y transferencias pueden ascender hasta un 25% del total [HBR1]).

Las entidades bancarias tienen que reaccionar y deben que hacerlo pronto y de la forma adecuada. El objetivo consiste en transformar el modelo de negocio y asentarse allí donde otras empresas se están afianzando. Pues, no tendrá más éxito el que ofrezca un mejor producto, si no el que ofrezca el producto que el cliente demanda y de la forma que lo reclama. Se requiere abandonar el clásico enfoque de producto y centrarse en conocer y satisfacer las necesidades y expectativas reales del cliente.

La solución pasa, invariablemente, por cambiar la mentalidad y las prácticas imperantes en el sector. Para ello, será necesario acometer un análisis holístico de su modelo de negocio. Hay que reevaluar todos los aspectos de su funcionamiento: sus principios y valores, su estructura, las prioridades y el enfoque y cómo todo ello se proyecta hacia el cliente.

En lo concerniente a las oficinas, múltiples estudios abogan por la digitalización total del sector. Lo que implicaría un cierre masivo de oficinas. Sin embargo, la inversión dedicada a la creación de la red de oficinas ya está amortizada. Proceder al cierre de la mayoría de oficinas haría que las entidades perdiesen uno de sus actuales puntos fuertes: la cercanía y el trato cara a cara con el cliente [Kurt]. Lo que es, sin lugar a dudas, la mayor carencia de los nuevos competidores del sector. Recientemente, Accenture, realizó otra encuesta al respecto [ACC2]. En ella se preguntaba, entre otras cosas, cómo era la visión de la gente respecto de las oficinas y su utilidad para ellos. De los resultados se desprenden dos conclusiones que dejan entrever la tendencia futura de uso de las oficinas:

- Un 85% de los encuestados creía que en el plazo de 5 años sus entidades dispondrían de un número igual o mayor de oficinas que actualmente.
- El 78% decía que usaría su red de oficinas igual o más a menudo que a día de hoy.

Para aumentar la tasa de retención y captación de clientes es necesario desarrollar una relación de confianza. Y, aunque esto deba hacerse a nivel global, el mejor punto de partida es reforzar el modelo de acceso al cliente desde el canal presencial. Sin embargo, el mantenimiento de la red de oficinas es muy costoso para las empresas del sector. Por tanto, la estrategia adecuada para este apartado debe basarse en la reorganización de las actividades de las oficinas, reducir de costes, y en redirigir su funcionamiento a un enfoque por y para el cliente.

Adicionalmente, se plantea otra dificultad añadida. La estructura interna de la mayoría de entidades, creada en tiempos pasados y engordada desde entonces, supone la mayor de las limitaciones. Un reciente estudio de PricewaterhouseCoopers [PWC1] cita las seis barreras internas más importantes para que un banco consiga su objetivo digital. Por orden de importancia, son éstas:

- Sistemas complejos y anticuados en el núcleo de su infraestructura y procesos tecnológicos.
- El marco regulatorio que afecta a sus actividades.
- Falta de presupuesto para abordar los cambios necesarios.
- Una cultura interna de organización poco proclive al cambio.
- Escasez de talento o conocimiento en las nuevas TIC.
- Falta de compromiso entre sus miembros directivos.

Las acciones que deban tomarse para resolver cada uno de estos puntos dependerán de las circunstancias intrínsecas de cada entidad. El presente documento, pretende dar parte de la respuesta que requiere el primer punto; especialmente en lo que a las oficinas se refiere. Pero, para poder abordar el diseño de una solución adecuada, antes se debe conocer la situación actual.

2.2 Contexto actual de una oficina bancaria

2.2.1 Entorno de oficina

El cambio que se debe acometer tiene que ser integral y ello implica cambiar también la propia estructura y distribución física de una oficina. Pues, siendo como son unos de los puntos de acceso preferenciales, deben transmitir una imagen de cambio y renovación. En los siguientes puntos, se procederá a una breve exposición de la situación actual de una oficina a nivel físico.

2.2.1.1 Tipología de servicios

La oficina comercial de banca ha sido desde hace decenas de años el principal punto de acceso de y al cliente. Este paradigma está cambiando con la digitalización del sector, pero las oficinas siguen teniendo una importancia capital en la concepción del modelo de negocio de muchos bancos.

Indistintamente de la entidad de la que hablemos, podríamos establecer que una oficina de banca comercial se organiza en torno a tres funciones básicas:

- **Servicios de caja**: son las operativas del día a día del cliente medio, como pueden ser: ingreso/retirada de efectivo, pago de impuestos o recibos, consulta del estado de una cuenta, etc.
- **Servicios avanzados y comerciales**: engloban actividades más heterogéneas como puedan ser: asesoría personalizada, captación de clientes y contratación de productos. Aquí se ubican las fuerzas de venta y forman el verdadero núcleo de negocio de una oficina.
- **Autoservicios**: se compone normalmente de cajeros y demás dispositivos pesados que posibilitan que el cliente efectúe operativas básicas, posiblemente fuera de horario comercial, sin supervisión de un empleado. El catálogo de servicios ofrecidos por los autoservicios a menudo duplica a los de caja.

2.2.1.2 Hardware

Una oficina tipo necesita de diversos dispositivos para poder ofrecer los servicios que le son requeridos. Éstos se pueden categorizar de la siguiente manera:

- **Fundamentales**: en este grupo se recoge la maquinaria básica que permite a una oficina abrir. Se compone principalmente de los ordenadores, con su periferia básica (teclados, ratones y pantallas) y el hardware de red.

- **Dedicados**: esta categoría engloba todo elemento periférico que se encuentra a disposición de un único puesto de banca. Como por ejemplo: teléfonos, impresoras financieras, lectores de cheques, chips, bandas magnéticas o biométricos, escáneres, dispositivos de captura de firma (tabletas o bolígrafos digitales), recicladores[1] y dispensadores[2] de divisa, contadoras de billetes, etc.
- **Compartidos**: son aquellos aparatos, a menudo de gran formato, que son compartidos por el personal de la oficina como pueden ser, por ejemplo, las impresoras en red.
- **De valor añadido**: son aparatos de naturaleza muy particular cuyo fin es el de servir a un determinado propósito que a menudo es opcional y no tiene por qué estar relacionado con los servicios ofrecidos al cliente. Ejemplos de este tipo de máquinas pueden ser los tótems de asignación de turno, detectores de metales, pantallas publicitarias, regletas eléctricas programables, etc.

2.2.1.3 Topología de una oficina

Dichos servicios se distribuyen físicamente en las oficinas de una forma bastante parecida, independientemente de la entidad de la que hablemos. Por lo que podemos establecer una topología de oficina, que se representa en la Ilustración 1:

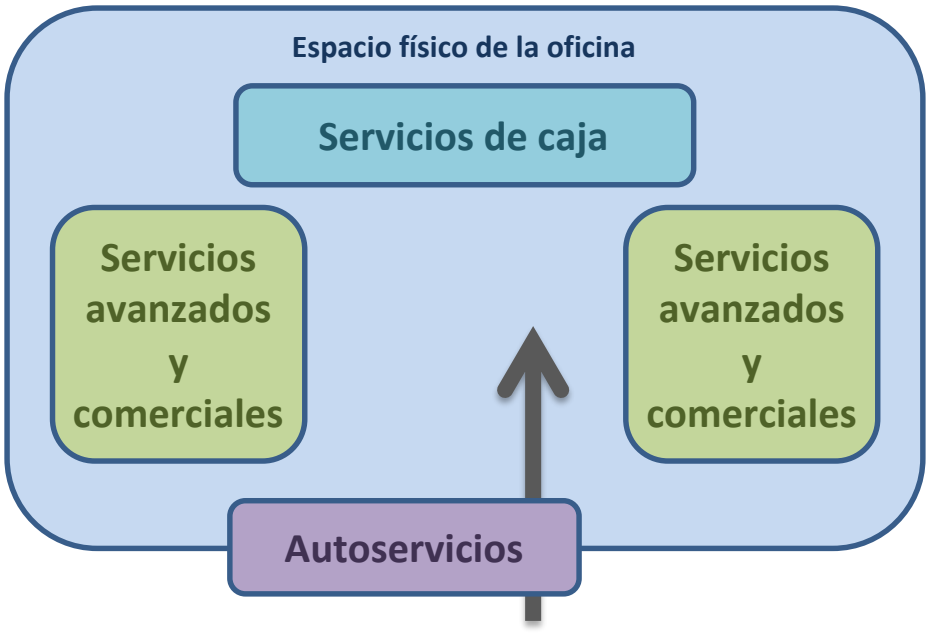

Ilustración 1 - Topología actual de una oficina

[1] Un reciclador de divisa es un dispositivo acorazado que permite ingresar y dispensar billetes de una o más divisas directamente desde un puesto de oficina.
[2] Un dispensador es una máquina similar a un reciclador pero carece de la capacidad para reconocer y validar la divisa. Por este motivo, no permite ingresar efectivo.

Como se puede apreciar en el anterior diagrama, los autoservicios se ubican fuera y/o a la entrada de la oficina para facilitar su uso fuera de horario. El área de servicios de caja se encuentra al fondo en una zona donde tenga visibilidad desde la entrada. En la periferia restante se sitúa el espacio dedicado a servicios comerciales donde se dispone de un espacio más flexible para colocar impresoras multifuncionales de gran formato, escritorios y demás mobiliario de oficina, etc.

2.2.1.4 Puesto de trabajo

Como veremos a continuación, la función de un puesto de trabajo define su composición y apariencia. En base a ello, podríamos dividirlas en dos tipos:

- **Puesto de caja**: menos abundante que el comercial, se configura para dar soporte a los servicios de caja. Para ello, dispone principalmente de un ordenador, un reciclador de divisa, una impresora financiera, un lector de cheques, otro de códigos de barras y un dispositivo de captura de firma. Es frecuente encontrarlos ubicados tras un mostrador o parapeto (a veces incluso una pantalla acristalada) que separe al cliente del empleado de caja. Con esto se consigue reducir el riesgo de hurto y/o robo, dotar al puesto de presencia y esconder la periferia.
- **Puesto comercial**: visualmente son puestos más convencionales formados por un escritorio de oficina, dotado de una mayor o menor imagen corporativa, ordenador, escáner, dispositivo de captura de firmas e impresora (bien dedicada o compartida).

2.2.2 Infraestructura y modelos de despliegue

Para que los empleados puedan desarrollar sus funciones disponen de un programa que denominaremos Escritorio de Trabajo (EdT). Dicha herramienta de trabajo suele mostrarse a pantalla completa y provee los medios necesarios para la realización de las labores tanto comerciales como de caja del día a día. Este tipo de aplicaciones son muy específicas de cada entidad pues éstas diseñan su propia aplicación a medida ajustada a sus necesidades, prioridades e infraestructura. Pueden haberse desarrollado íntegramente o depender de módulos de terceros para operaciones concretas. Sin embargo, a día de hoy, la mayoría de EdTs se pueden englobar en dos modelos fundamentales de funcionamiento y despliegue que condicionan la infraestructura y procesos subyacentes.

2.2.2.1 Escritorio local

En este caso el EdT es una aplicación convencional de escritorio, programada normalmente en .NET o Java, que se ejecuta en el puesto del usuario. Como se puede apreciar en la Ilustración 2, la periferia de oficina se conecta directamente al puesto y es

accesible desde el EdT de forma nativa, lo que simplifica su manejo. La oficina dispone habitualmente de una línea principal de comunicaciones y otra de respaldo. Para que los puestos de la oficina tengan conectividad con el CPD, se establece un canal seguro (vía VPN, VPLS o similar) entre el router de la oficina y el CPD. De esta forma, se establece una macro LAN virtual compuesta por todas las subredes de cada una de las oficinas y parte del CPD. A partir de este punto, el EdT puede comunicarse con servidores de aplicaciones, bases de datos, almacenamiento en red, sistemas host, etc. del CPD.

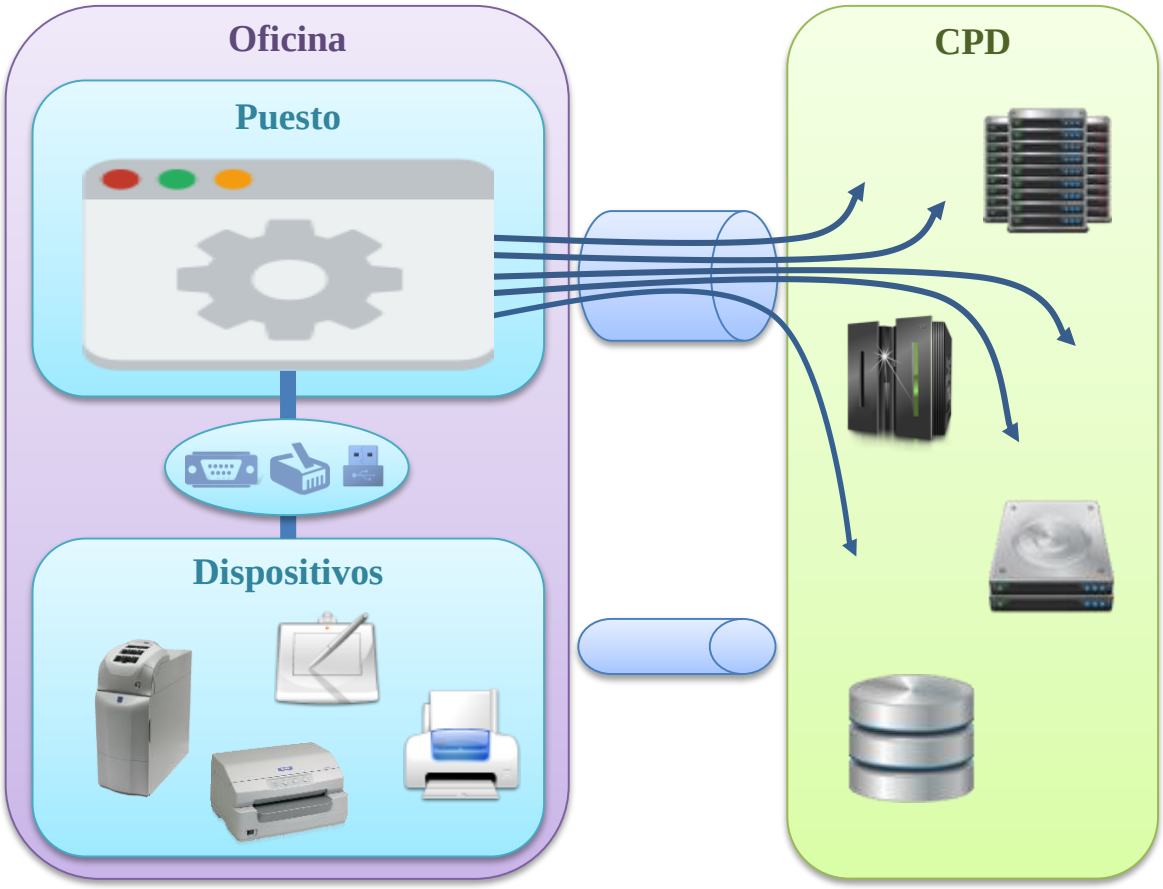

Ilustración 2 - Arquitectura de Escritorio de Trabajo local

Este modelo de despliegue se ajusta a una arquitectura Cliente–Servidor, por lo que hace un uso convencional de la red de comunicaciones en los intervalos de operación de la oficina. Sin embargo, el EdT suele ser una aplicación compleja y con muchas dependencias por lo que, bajo este modelo de arquitectura, la distribución y versionado de una aplicación de este tipo y tamaño no son tareas triviales. Además, las distribuciones son costosas en tiempo y recursos por lo que una marcha atrás de una versión errónea no siempre es viable. Por tanto, deben dársele especial importancia a las fases de entornos previos a producción pues un error en una distribución puede conllevar días o incluso semanas de inoperatividad.

2.2.2.2 Escritorio virtualizado

Este modelo es similar a la arquitectura clásica de banca del siglo XX en la que una máquina que actuaba como terminal ejecutaba una sesión remota en un mainframe del

CPD. Actualmente, la virtualización provee unas funcionalidades parecidas pero de forma mucho más potente. Como se puede apreciar en la Ilustración 3, en este modelo el EdT se ejecuta en un servidor del CPD y se sirve, virtualizado, en el puesto.

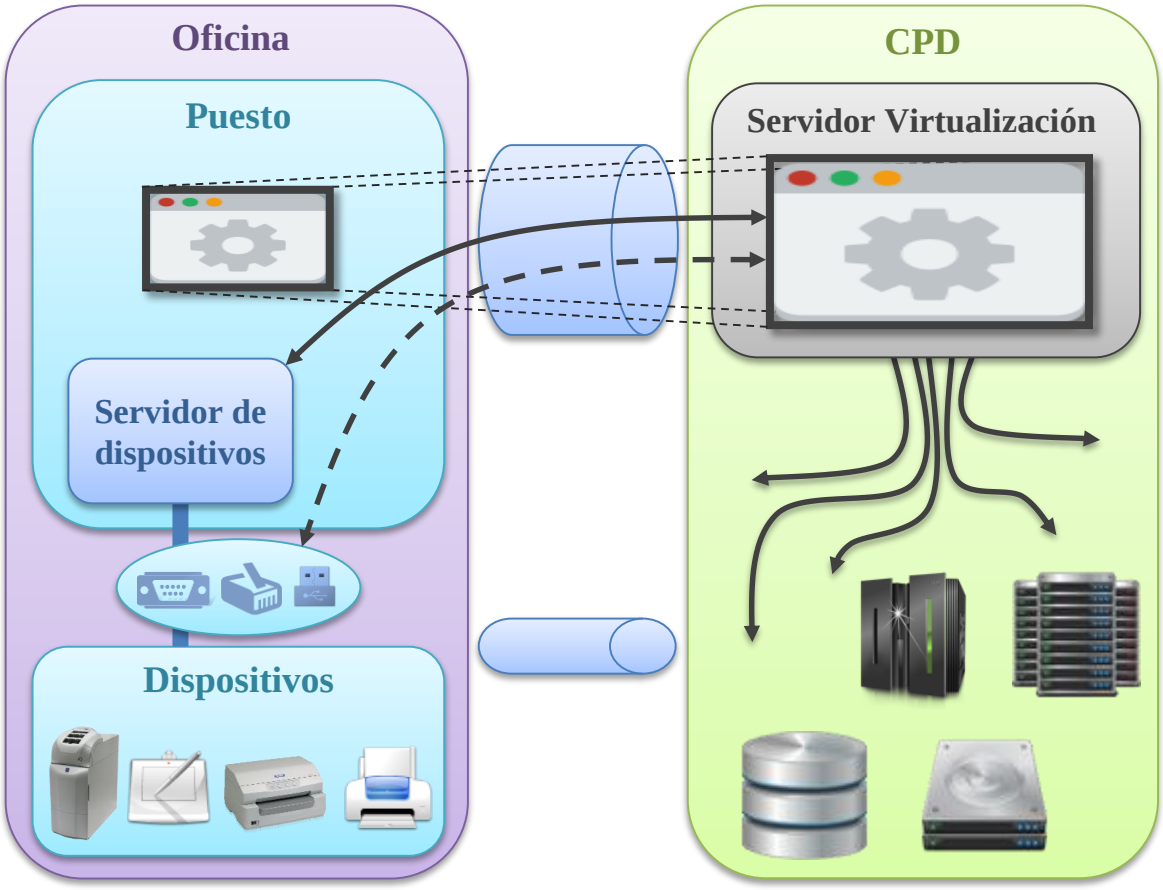

Ilustración 3 - Arquitectura de Escritorio de Trabajo virtualizado

Al ejecutarse el EdT dentro del CPD su acceso a servidores de aplicaciones, bases de datos, almacenamiento en red, sistemas host, etc. del CPD es casi inmediato. Sin embargo, la periferia permanece en el puesto. Por lo que su acceso se hace de forma remota bajo un protocolo cliente-servidor, estandarizado o hecho a medida. Según el sistema de virtualización que se utilice, para ciertos dispositivos es posible utilizar una tecnología denominada Canal ICA [ICA1]. Este protocolo permite la creación de un canal virtual que puede transportar, entre otros, las señales de bajo nivel de los periféricos a la aplicación que se ejecuta en el servidor. Sin embargo, y debido principalmente a las latencias de red, este mecanismo no se puede utilizar en todas las situaciones y no es completamente fiable con ciertos dispositivos [ICA2]. Cualquiera de estas estrategias de acceso a periféricos implica una latencia y utilización de ancho de banda relativamente elevados.

Adicionalmente, al tráfico derivado del uso de la periferia hay que tener en cuenta el derivado de la virtualización y los procesos puramente productivos. Consecuentemente, en este tipo de arquitecturas, las líneas de comunicaciones suelen tener un factor de utilización mucho más alto durante las horas de apertura de las oficinas. Por ese motivo, es necesario

aplicar políticas de priorización de tráfico para garantizar que la virtualización, entre otros, funcione correctamente. Esto, a su vez, puede derivar en que ciertos componentes del EdT sufran latencias muy elevadas, tasas de transferencia reducidas o incluso microcortes o descarte de paquetes.

Contrariamente a lo que ocurría en el caso anterior, la instancia del EdT que se ejecuta en este esquema es única para toda la red de oficinas y se encuentra disponible dentro del CPD. Por ello, su distribución, versionado y mantenimiento son mucho más sencillos que en el caso anterior.

2.2.3 Debilidades del modelo actual

De los apartados anteriores se desprende que existe un inconveniente en la distribución de áreas funcionales. Si bien dicha distribución es sencilla y limpia, confiere una presencia excesiva al puesto de caja. Sin embargo, no existe una relación de proporcionalidad entre la importancia prestada a dicho puesto y el beneficio efectivo que reporta a la entidad. Así mismo, el mobiliario de oficina, el tipo de equipos utilizados y su emplazamiento son muy convencionales, dificultan la movilidad y no aportan imagen de marca.

En otro orden de cosas, la abundante presencia de periferia dedicada supone un punto débil a tener en cuenta. Por un lado, la adquisición y mantenimiento de todos esos equipos implica un alto coste. Por otro, el escenario apenas contempla la opción de compartir dispositivos. Esto deriva en que, ante un hipotético caso de indisponibilidad de un periférico, se comprometa la productividad de un empleado y, posiblemente, la de la oficina misma.

En cuanto a la infraestructura que soporta los procesos operativos, cada arquitectura tiene sus pros y sus contras. Sin embargo, ambos modelos presentan ciertos riesgos de funcionamiento y eficacia en el medio-largo plazo para cualquier organización que pretenda incorporar procesos TIC modernos. El primer esquema lastra a los equipos de desarrollo por el uso de tecnologías pesadas que probablemente no se integren correctamente con una experiencia de usuario adecuada a los tiempos que corren. Así mismo, impone tiempos y plazos de distribución muy elevados que dificultan la implantación ágil de nuevas soluciones o correctivos. Por último, los periféricos deben están físicamente conectados a la máquina donde corre el EdT, por lo que hace muy complejo, si no imposible, la compartición de dispositivos o un cambio en la disposición y tipo del entorno hardware utilizado.

El segundo modelo, al depender de una única copia del EdT (presente en el CPD), solventa parte de las limitaciones anteriores. Sin embargo, es una solución de compromiso pues implica una gran inversión en servidores, licencias y equipamiento de red, añade complejidad a la arquitectura y agrava la dependencia respecto de los sistemas de comunicaciones de las oficinas y el CPD.

2.3 Retos y objetivos del cambio

Como ya se comentó en la introducción, el objetivo principal de este proyecto es el de rediseñar el concepto de oficina bancaria, actualizándolo al contexto actual, modelando una imagen y personalidad diferentes y fundamentando dicha transformación en una profunda actualización tecnológica. Para ello, habrá que reevaluar muchos de los pilares básicos del diseño de una oficina y modificarlos hasta alcanzar una separación conceptual respecto al resto de la competencia. Por ello, el principal reto es **conseguir diferenciación.** Y eso sólo es posible teniendo en cuenta la diferenciación como componente vertical desde el principio del proceso y como constante en el diseño.

Derivado del punto anterior podemos observar que tras años sin cambios en un modelo comúnmente interiorizado por el público en general, el cliente no percibe un valor añadido en la disponibilidad de oficinas. Es necesario devolver el foco de interés a la oficina como punto de acceso a la entidad. Esto nos lleva a otro objetivo importante: **redefinir la forma de aproximación e interacción con el cliente**.

Con la llegada de Internet la forma de interactuar con el cliente, en el escenario de una oficina, se está quedando obsoleta. La tasa de penetración de las TIC en la atención al cliente es realmente baja. Este hecho hace que, inconscientemente, el cliente no valore el coste dedicado a mantener una oficina funcionando ni el beneficio que puede obtener de ella. Por ello, otro factor determinante es el de **modernizar la imagen de la oficina**. Y para lograrlo, un requisito imprescindible es el de **actualizar la plataforma tecnológica**. Sin un entorno que soporte los nuevos retos tecnológicos es imposible lograr los objetivos impuestos.

Adoptar la religión de los trending topics y los nightly builds es sencillo; sin embargo, evolucionar un modelo de oficina bancaria no es una tarea trivial ni que pueda ser abordada de la noche a la mañana. Por ello es necesario mantener un alto nivel de **compatibilidad con los procesos actuales** para garantizar una transición no disruptiva.

Un requerimiento mucho más pragmático es el de **acentuar el puesto de ventas de la oficina bancaria**. Es decir, focalizar la razón de ser de la oficina en la productividad y eficiencia en términos de negocio: hay que maximizar el beneficio a la vez que se minimiza el coste de conseguirlo. Es por ello inevitable hablar también de un **ahorro de costes frente al modelo actual**.

Por último y precisamente porque se van a destinar muchos recursos a la creación e implantación de esta renovada filosofía de banca, hay que **garantizar la sostenibilidad de la nueva plataforma** a medio y largo plazo. Y ello implica abordar el diseño del proyecto desde un punto de vista que asegure su ciclo de vida y evolución.

Capítulo 3

Planteamiento de la solución

3.1 Nuevo paradigma de oficina

Como ya se comentó anteriormente, en los últimos años están apareciendo nuevas formas de ofrecer servicios financieros. Este cambio se apoya en una alta componente de frescura, ubicuidad, tecnología y sencillez. Esta innovadora forma de entender la banca está calando hondo entre los consumidores, especialmente los jóvenes, creando una demanda de servicios similares. Esta situación ha logrado devaluar, por simple comparación, la imagen que los clientes tienen de sus entidades tradicionales. Las entidades bancarias deben, pues, actuar para recuperar la confianza y el interés del cliente.

Para conseguirlo, es necesario cambiar la propia filosofía que rige el funcionamiento actual de las entidades por una visión renovada que cumpla con esta reciente forma de entender y consumir los servicios financieros. Por tanto, para que el cambio sea efectivo, es obligado adoptar un enfoque holístico de la situación. Las modificaciones deben afectar a todos y cada uno de los aspectos del día a día en su conjunto.

Llegados a este punto, y una vez estudiada la situación actual y establecidos los objetivos que deben ser satisfechos, el siguiente paso es dar forma a un nuevo modelo de oficina. Procedamos.

3.1.1 Disposición del espacio de atención al cliente

El motor de cambio ha de ser la orientación y cercanía al cliente como seña de identidad, y la solución debe diseñarse con esta idea como eje vertical en el que basar todas las decisiones a tomar, incluida la disposición del área de atención al cliente.

Ilustración 4 - Puesto Cocoon de BBVA

Bajo este principio, se ha de concebir un nuevo entorno en el cual el cliente se sienta parte de las operaciones, que pueda seguir el proceso en primera persona y hablar con los empleados de tú a tú. Por poner un ejemplo válido, podemos hacer referencia a la Ilustración 4 que muestra un tipo de puesto ideado por BBVA llamado Cocoon [BBVA1]. Su disposición está ideada para que cliente y empleado se sienten lado a lado de tal manera que el cliente pueda ser guiado y aconsejado de forma transparente y versátil.

3.1.2 Equipos y movilidad

La disposición del espacio de atención al cliente y sus los puestos determina en parte la naturaleza de los equipos a utilizar. En este aspecto, es necesario dar un salto cualitativo. Por lo que los equipos utilizados deberán tener una alta componente de movilidad y pantalla táctil. El objetivo es facilitar el movimiento y la interacción empleado-cliente en un contexto y postura mucho más naturales y transparentes.

En este modelo de oficina se deben considerar equipos cuya huella visual no sea excesiva pero sí atractiva. El objetivo es terminar con los entornos de trabajo atestados de papeles y objetos. Centrar la atención del cliente únicamente en el terminal y el empleado para que la información pueda transmitirse de la manera más eficiente posible.

Esta elección aporta una serie de ventajas adicionales e inherentes al concepto de movilidad. El hecho de que un empleado disponga de un teléfono móvil y un equipo portátil incrementa la continuidad del trabajo, la productividad y la agilidad, reduce los tiempos de respuesta, los costes y proyecta imagen de innovación.

Así pues, las opciones se reducen a ordenadores 2 en 1 con teclado encastrable, similar al de la Ilustración 5, como equipos de referencia para el día a día del empleado, ya que su tamaño, diseño y versatilidad encajan a la perfección con los requisitos anteriormente citados. A día de hoy, la práctica totalidad de fabricantes dispone de modelos con estas características[1].

Ilustración 5 - Ejemplo de ordenador 2 en 1

3.1.3 Isla de dispositivos

Las características tanto de los puestos como de los equipos elegidos condicionan enormemente el uso de periféricos. El uso de estos dispositivos es necesario, pero se puede idear una forma mejor de disponer de ellos que sencillamente enchufándolos al ordenador en cuestión. Con este objetivo se han ideado las Islas de Dispositivos (IdD).

Una IdD es una pieza de mobiliario cuya función consiste en alojar toda la periferia necesaria y hacerla accesible a todos los terminales de la oficina. La configuración de periféricos en ella contenidos podrá ser variable en función de las necesidades y del área en que esté ubicada. Se pueden diseñar de varios tamaños y ubicar tantas como sea necesario de acuerdo al tamaño y volumen de trabajo de la oficina.

Esta configuración aporta múltiples beneficios más allá de los puramente estéticos. Hacer que la periferia sea accesible desde a través de las IdDs implica que los dispositivos

[1] Por mencionar algunos modelos, se podría optar por la Microsoft Surface Pro, Lenovo Helix, Asus Transformer, HP Split 2x o incluso el Sony VAIO Duo.

sean compartidos por todos los empleados, independientemente de en qué parte de la oficina se encuentren. Y esto a su vez conlleva una serie de beneficios:

- Se puede reducir la duplicidad de periféricos lo que supone un importante ahorro directo de costes tanto en adquisición como en instalación y mantenimiento.
- También se puede conseguir un ahorro de costes indirecto al disponer de dispositivos cuya contraparte con capacidades de compartición son notablemente más caras. Como por ejemplo impresoras o escáneres.
- El hecho de que un terminal pueda hacer uso de la periferia de más de una isla aumenta la disponibilidad y, por tanto, la productividad.

3.1.4 Áreas funcionales

El espacio de una oficina debe ser rediseñado con el objetivo de priorizar funcionalmente las distintas zonas. La distribución propuesta de éstas se puede ver en la Ilustración 6.

Ilustración 6 - Áreas funcionales de una nueva oficina

Área de cliente

Se debe disponer de un área específica en la que todos los elementos se configuren para dar servicio y cultivar la relación con el cliente. Esta es la zona en la que hay que poner más esfuerzo pues representará el punto de acceso de las oficinas y, por tanto, la cara vista del canal presencial de la entidad. Por ello, es importante destacar que, para transmitir la imagen deseada, la estética y disposición de todos los elementos de la composición han

de ser acordes con unas líneas básicas que transmitan cercanía con el cliente, simplicidad, accesibilidad y limpieza visual como principales rasgos de diseño. Bajo estas premisas, se debe contar con un espacio lo más despejado posible, con asientos confortables y, si fuese posible por tamaño y volumen de clientes, con puestos de atención al cliente.

Zona de trabajo

Este será el espacio de trabajo de los empleados. Puede estar a la vista o en una sección independiente pero siempre aislada del área de cliente. Tendrá la disposición y configuración necesarias para realizar las labores de back-office, incluida una IdD. Típicamente constarán de escritorios convencionales integrados en la estética general de la oficina.

Área privada

En cierto tipo de oficinas puede ser necesario disponer de una zona con un ambiente de mayor privacidad y distinción positiva que la de cliente. Para cubrir estas necesidades, se ha ideado el área privada. Esta zona se compone de una o más salas donde poder atender las necesidades que por la naturaleza de la operación o del cliente necesiten de las características de este espacio. Su diseño deberá seguir mismas las líneas maestras que el resto del conjunto. Asimismo deberá disponer de una IdD.

Perímetro de autoservicios

En el exterior y el perímetro del área de cliente se ubicarían los autoservicios de nueva generación que detallaremos más adelante.

3.1.5 Nueva generación de autoservicios

Todos los puntos anteriores conforman un ecosistema que hace especial énfasis en la creación de un conjunto de servicios avanzados y comerciales orientados hacia el cliente. Sin embargo, los clientes también demandan la posibilidad de acceder a servicios de caja para sus operaciones del día a día. Con el objetivo de dar respuesta a esta demanda se creará una nueva generación de autoservicios que posibilite el acceso a un mayor catálogo de servicios de forma autónoma.

Estas máquinas se ubicarán en la periferia del área de cliente. Deberán contar con molduras o muebles a medida que se integren en la estética del resto de la oficina. Además, sería interesante contemplar el diseño desde el punto de vista de la comodidad del cliente. Que el usuario disponga de un asiento, un pequeño parapeto que aporte privacidad o una zona donde depositar sus enseres son elecciones interesantes. También se puede barajar la opción de incorporar ciertos accesorios como, por ejemplo, un paragüero o una papelera.

Algunos ejemplos del tipo de operaciones que se pueden migrar a los autoservicios serían: los ingresos y retiradas de efectivo, el pago de impuestos, multas y recibos, ordenar

transferencias o donaciones, ingreso de cheques, etc. También se debe prestar especial atención a la interfaz de usuario, para que ésta sea más acorde con las interfaces actuales a las que el usuario está acostumbrado. Asimismo, se debe recurrir al uso de nuevas tecnologías y procesos como pueden ser NFC (Near Field Communication), acceso por OTP (One Time Password), pantallas personalizadas de inicio de sesión, preselección de operaciones más habituales, etc.

Este punto se encuentra muy avanzado actualmente. Multitud de entidades a nivel mundial soportan algunos de los servicios citados en sus autoservicios. Sin embargo, el despliegue de servicios no es suficiente y, en muchos casos, no se ha implementado de la manera correcta. Los fallos más comunes son la falta de servicios, mala ubicación del autoservicio, escaso número de ellos, interfaz, apariencia y medios de acceso desfasados, pobre usabilidad, etc. Si se propone derivar a un usuario de un puesto de caja a un autoservicio, éste debe sentirse cómodo y natural durante la operación y satisfecho tras ella.

3.1.6 Nuevas prácticas y servicios

Transformar la filosofía, aspecto y funcionamiento de las oficinas no servirá de nada sin una plétora de innovadores servicios que atraigan a los clientes. Bajo estas premisas, y con la idea principal de reenfocar el negocio hacia el cliente, se plantea un catálogo de nuevas prácticas y servicios de recomendado cumplimiento tanto dentro como fuera de las oficinas.

En primer lugar, hay que resaltar una obviedad: si queremos aprovechar la ventaja de la cercanía y el trato humano que ofrecen las oficinas, éstas deberían adaptar su horario de acuerdo al ritmo de vida de los consumidores potenciales. Actualmente, existe una patente indisponibilidad de franjas de servicio atractivas y acordes con las rutinas de los clientes. **Es necesario alargar o reorganizar los horarios de apertura**. Una propuesta equilibrada sería, por ejemplo, la de abrir dos o tres días en jornada de tarde hasta las nueve o diez de la noche. Otra opción, no necesariamente excluyente con la anterior, sería la de abrir los sábados por la mañana; aunque sólo fuese en el fin de semana con más afluencia del mes.

También es importante, **desligar a un cliente de su oficina**. Teniendo en cuenta la deslocalización que aportan las soluciones tecnológicas y otros medios de interacción, raya el absurdo que un cliente deba acudir única y exclusivamente a una oficina en concreto para realizar ciertas tareas o gestiones.

Por otro lado, el actual **catálogo de productos** es grande, poco novedoso, algo confuso y, a veces, presenta demasiadas condiciones, contrataciones cruzadas o letra pequeña. Esta misma situación ya ha ocurrido en otros sectores como el las telecomunicaciones. El cliente demanda sencillez y transparencia, y el catálogo de productos debe ir acorde con estas premisas. Un número de productos contenido y de características sencillas hará más fácil

su explicación por parte del personal de la entidad, y permitirá concebir nuevas formas de presentación y publicitación al cliente.

Uno de los servicios que más valor puede aportar a una entidad es el de ofrecer un **gestor personal**. Mediante este servicio, los clientes podrían contactar por teléfono, correo, chat o video conferencia con un grupo fijo de gestores comerciales. El número de gestores asignados a un cliente debe ser reducido, permanente y deben tener horarios fijos repartidos en distintos días. El objetivo es que el cliente pueda establecer una relación personal y de confianza con sus gestores más habituales. El gestor principal debería ser uno ubicado en una oficina de la elección del usuario, con el fin de que también pueda ser accesible presencialmente. Para cubrir toda la franja horaria puede ser necesario recurrir a la deslocalización. Actualmente ya existen, en algunas entidades, iniciativas encaminadas a prestar este servicio.

El conocimiento de las necesidades reales de los consumidores determina el éxito en las acciones de venta de cualquier empresa. Asimismo, una comprensión suficiente de la base de clientes permite conocer el estado real de la demanda. Y, con ello, se puede maximizar del ratio de beneficio-cliente mediante la distribución óptima de las inversiones, la personalización de productos, la segmentación de campañas de venta, etc. Con tal fin, se utiliza lo que se denomina **Big Data** [WIKI01] [BIG1] [IBM1]. Éste es un concepto muy amplio que engloba una serie de técnicas de recogida, filtrado, ordenación y análisis de datos de diferentes tipos y fuentes que, a priori, podrían parecer no relacionados.

Cada vez más a menudo, la gente basa sus elecciones de compra de productos en la información recogida mediante una forma de inteligencia colectiva denominada crowdsourcing. El tiempo dedicado a recabar información suele ser directamente proporcional al coste del bien a adquirir. El correcto uso de Big Data por una entidad con una gran base de clientes hace posible ofrecer un **servicio de asesoría y ayuda a la compra** de cualquier tipo de bien. Dicha asistencia puede ir condicionada o no a la contratación de un producto de financiación. Otros posibles usos relacionados con el Big Data serían la aplicación de descuentos o las ventas cruzadas entre otros.

A día de hoy, y cada día más, la gente utiliza diariamente distintos servicios en la nube como pueden ser el correo, almacenamiento, calendario, etc. Todo ello disponible de forma ubicua desde sus ordenadores y teléfonos. Si una entidad pretende pasar a formar parte de la vida diaria de una persona debe, forzosamente, **integrarse con servicios de terceros**. Comunicaciones por correo electrónico con su gestor, envío de documentación a servicios de almacenamiento, comunicaciones por vías alternativas, concertación de citas, etc. son sólo algunas de las opciones.

Pese a ser un nicho estable y con miembros claramente dominantes, los **sistemas de transferencia peer to peer** ofrecen a las entidades la posibilidad de establecerse a nivel mundial como medio de pago o transferencia por Internet. Y tener acceso a un mayor número de clientes potenciales para el resto de sus servicios.

Otro punto muy importante reside en las **aplicaciones dedicadas y experiencia de usuario**. Respecto al primero, cada uno de los servicios hasta ahora expuestos debe poder ser accesible ya sea mediante un navegador, por teléfono móvil o con el uso de asesoría presencial en la oficina. En cuanto a lo segundo, la experiencia de usuario debe ser homogénea, sencilla, vistosa y con una alta usabilidad.

Capítulo 4

Plasmando el nuevo modelo

4.1 Introducción

Llegados a este punto, ya tenemos una visión clara de cuál es la solución que se persigue. En este capítulo se detallará el diseño de una plataforma tecnológica que haga posible implementar la solución planteada e incorporar futuros servicios de forma ágil. Para conseguir que dicha solución satisfaga los objetivos impuestos, es necesario tener en cuenta todos los componentes que forman parte de ella y cómo se relacionan unos con otros.

Dada la naturaleza del escenario existente, necesitaremos disponer de múltiples piezas tanto dentro como fuera del CPD. Algunas de ellas, incluso estarán fuera del área de control e influencia de la entidad. Sea cual sea la naturaleza de estas piezas, deben poder comunicarse unas con otras de forma óptima a través de redes de comunicaciones de capacidades diversas. Por último, será necesario proveer acceso al hardware de periferia de la oficina.

Como se irá detallando más adelante, la nueva arquitectura hace especial hincapié en movilidad, dinamismo, ubicuidad, uso de servicios de terceros y compartición de dispositivos. Todo ello mediante en el uso de una arquitectura de servicios web que, a su vez, será soportada sobre una infraestructura de nube híbrida corporativa. Estas ideas se irán presentando en las siguientes secciones, junto con una pequeña explicación de las tecnologías que las soportan. El objetivo es comprender las herramientas existentes y conocer sus debilidades y puntos fuertes. Pues todo ello ha influido a la hora determinar la solución al escenario propuesto.

4.2 Arquitectura de servicios

En los últimos años, la globalización de los mercados y compañías, han propiciado un cambio en la forma de entender las relaciones empresariales, principalmente en el ámbito tecnológico. Por una parte, se ha modificado el hábitat empresarial de tal forma que hoy en día ninguna empresa ni proceso de negocio está aislado. Esto se debe en gran medida a que para poder desempeñar sus funciones necesita manejar información de y hacia sus proveedores, socios, clientes, departamentos o empleados. Y todo ese manejo de información se desarrolla a través de redes y sistemas que se extienden más allá de su propiedad y control.

Por otro lado, invertir en TIC ya no se trata de construir grandes infraestructuras que posibiliten la comunicación; si no de proveer nuevos usos de la información, de una manera eficiente, barata y atractiva. Hoy en día no es mejor la empresa que sea capaz de crear valor añadido, si no la que sea capaz de crearlo más rápidamente; pues la innovación es susceptible de ser copiada.

A simple vista, componer y coordinar un proceso de negocio puede parecer una labor sencilla. Sin embargo, debe tenerse en cuenta la heterogeneidad existente en las infraestructuras actuales (sistemas operativos, lenguajes de programación, bases de datos, protocolos de comunicación, implementaciones hardware, etc.). Esta gran cantidad de plataformas disímiles hace virtualmente imposible abordar la cuestión con un enfoque frontal. La solución radica, básicamente, en buscar el mínimo común denominador a todas ellas y hacerlo evolucionar hasta cubrir las necesidades requeridas. Este es, precisamente, el escenario donde las *arquitecturas de servicios* toman la relevancia que se les confiere hoy en día.

El uso de arquitecturas de servicios a nivel empresarial aporta múltiples beneficios como pueden ser: rapidez y sencillez en la implantación de nuevos procesos, apertura a terceros, estandarización, integración de tecnologías heterogéneas, modularidad, etc. Actualmente existen dos grandes familias de arquitecturas disponibles: SOAP y REST cada una de ellas con sus ventajas e inconvenientes. Para poder entender la decisión acerca de

cuál usar y en qué contexto, es necesario que primero revisemos las fortalezas y debilidades de cada una.

4.2.1 Arquitectura SOAP

4.2.1.1 SOA (Service Oriented Architecture)

SOA es el concepto subyacente a la, mal llamada, arquitectura SOAP. SOA [WIKI02] no es una arquitectura como tal. En su lugar podríamos hablar de un conjunto de requerimientos que, una vez implementados, permite integrar un proceso de negocio de principio a fin de una forma ágil. SOA no es, por tanto, una tecnología en sí misma si no una estrategia de negocio implementada sobre una base tecnológica. Es habitual referirse a las arquitecturas SOA como SOAP (Simple Object Access Protocol) pues es precisamente ésta tecnología su máximo representante. En la siguiente sección estudiaremos con más detalle el protocolo SOAP.

La unidad operativa mínima en este tipo de arquitecturas son los servicios web (WebService). Distintos servicios han de ser capaces de interactuar entre sí conforme a unas reglas para construir procesos autónomos que puedan agruparse para crear, a su vez, procesos de negocio globales. Los requisitos planteados por SOA para dichos servicios [WIKI02] son los siguientes:

- Todos los servicios deben ser diseñados pensando en su *reusabilidad*, ya sea dentro del mismo proceso, dominio de aplicaciones de la empresa o incluso del dominio de uso público.
- Para ser utilizados deben *poder ser descubiertos* por otros agentes.
- Todo servicio debe proporcionar una interfaz o *contrato formal* en el cual consten: el nombre del servicio, forma de accederle, las funcionales que ofrece y la naturaleza de los datos (tanto de entrada como de salida) de cada una de ellas. De esta manera, todo consumidor del servicio, accederá a él mediante su contrato, logrando así la independencia entre el consumidor y la implementación del propio servicio.
- Los servicios tienen que ser independientes los unos de los otros. Para conseguir ese *bajo acoplamiento,* se accederá a ellos a través de su contrato, consiguiendo independencia entre los servicios productor y consumidor. Si se logra este bajo acoplamiento, entonces los servicios podrán ser reutilizables.
- Deben ser *componibles*. Es decir, deben ser construidos de manera que permitan a otros servicios genéricos hacer uso de ellos de forma transparente.

- Deben ser *autocontenidos*; de tal forma que no expongan la lógica contenida ni necesiten para su correcta ejecución de ningún otro dato más que los especificados en su contrato.
- Otro requisito es que sean *autónomos*, que encapsulen la lógica necesaria para su ejecución así como que tengan su propio entorno de ejecución. De esta manera el servicio es totalmente independiente.
- Un servicio debe ser *sin estado*: no debe guardar ninguna información sobre el consumidor del servicio ni sobre los parámetros que éste le facilite durante la invocación de sus funcionalidades. De ser necesario, dicha información deberá almacenarse en algún repositorio externo para evitar inconsistencias entre sucesivas llamadas de un mismo o distintos consumidores.

Como se desprende de la enumeración de requisitos, SOA enfatiza la definición de APIs (Application Program Interface) de servicio, encapsulando funcionalidades autónomas y descubribles. Por tanto, los beneficios directos, entre otros, de la aplicación de SOA son:

- Integración de tecnologías heterogéneas en un mismo sistema.
- Rapidez, sencillez y ahorro en el desarrollo e implantación de nuevos procesos de negocio.
- Facilidad para abordar modelos de negocio colaborativos o distribuidos.
- Posibilidad de apertura de los sistemas a terceros.
- La modularidad inherente garantiza actualizaciones de mínimo impacto.

4.2.1.2 SOAP (Simple Object Access Protocol)

SOAP es una especificación o protocolo para el intercambio de mensajes basados en XML [WIKI03]. Proporciona un modo de comunicación entre aplicaciones que se ejecutan en sistemas operativos diferentes, con diferentes tecnologías y lenguajes de programación. Dicha comunicación se puede llevar a cabo utilizando distintos protocolos y medios de transporte como pueden ser HTTP, SMTP, TCP, colas de mensajes, etc. Sin embargo, el más común de ellos es HTTP pues facilita el uso de SOAP desde navegadores y en entornos acotados por firewalls o proxys. A continuación se describen los elementos más importantes de SOAP [W3S1]:

soap:Envelope

Es la raíz de un mensaje SOAP por lo que engloba al resto de elementos. Y su función es básicamente la de empaquetar el mensaje. Su uso facilita la labor de detectar cuándo se ha recibido un mensaje SOAP completo. Es un elemento obligatorio dentro de un mensaje.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2001/12/soap-envelope" SOAP-
ENV:encodingStyle="http://www.w3.org/2001/12/soap-encoding" >

…

</SOAP-ENV:Envelope>
```

soap:Header

Este elemento es opcional y proporciona una manera de especificar ciertos requerimientos a nivel de aplicación. P.ej. se puede utilizar para especificar firmas digitales en servicios securizados.

```
<SOAP-ENV:Header>
     <t:Transaction xmlns:t="http://www.tutorialspoint.com/transaction/" SOAP-
ENV:mustUnderstand="true">5</t:Transaction>
</SOAP-ENV:Header>
```

soap:Body

Está comprendido dentro del *<soap:Envelope>* y es un elemento obligatorio. Su utilidad es la de englobar al contenido del mensaje SOAP propiamente dicho. Todas las etiquetas dentro de *<soap:Body>* son específicas de la aplicación y se definen en un esquema XML (`http://www.xyz.org/quotations` en el ejemplo inferior).

```
<SOAP-ENV:Body xmlns:m="http://www.xyz.org/quotations" >
     <m:GetQuotation>
          <m:QuotationsName>MiscroSoft</m:QuotationsName>
     </m:GetQuotation>
</SOAP-ENV:Body>
```

soap:Fault

Se utiliza para informar de un error durante el procesado de la petición. Se coloca dentro de *<soap:Body>* y puede contener las subetiquetas: *<faultCode>*, *<faultString>*, *<faultActor>* y *<detail>*.

```
<SOAP-ENV:Fault>
     <faultcode xsi:type="xsd:string">SOAP-ENV:Client</faultcode>
     <faultstring xsi:type="xsd:string">
          Failed to locate method (ValidateCreditCard) in class (examplesCreditCard) at
/usr/local/ActivePerl-5.6/lib/site_perl/5.6.0/SOAP/Lite.pm line 1555.
     </faultstring>
</SOAP-ENV:Fault>
```

El estándar SOAP [W3O1] define algunos mecanismos y etiquetas adicionales. Pero no se consideran relevantes para la comprensión del mismo. A continuación figuran ejemplos de una petición y su respuesta SOAP.

Petición SOAP

```
POST /Quotation HTTP/1.0
Host: www.xyz.org
Content-Type: text/xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2001/12/soap-envelope" SOAP-
ENV:encodingStyle="http://www.w3.org/2001/12/soap-encoding" >
        <SOAP-ENV:Body xmlns:m="http://www.xyz.org/quotations" >
          <m:GetQuotation>
               <m:QuotationsName>MiscroSoft</m:QuotationsName>
          </m:GetQuotation>
     </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Respuesta SOAP

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2001/12/soap-envelope" SOAP-
ENV:encodingStyle="http://www.w3.org/2001/12/soap-encoding" >
     <SOAP-ENV:Body xmlns:m="http://www.xyz.org/quotation" >
          <m:GetQuotationResponse>
               <m:Quotation>Here is the quotation</m:Quotation>
          </m:GetQuotationResponse>
     </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.2.1.3 WDSL (Web Services Description Language)

El contrato que define un WebService se puede empaquetar en un archivo WSDL [WIKI04]. En él podemos encontrar una definición del servicio consistente en su nombre y parámetros de entrada/salida. Los documentos WSDL deben ser entendibles para las entidades lógicas que consumen el servicio. Por ello se escriben en XML. A continuación se presenta el conjunto de etiquetas XML más importantes de WSDL [W3O2].

definitions

Es la raíz de todo documento, agrupa al resto de etiquetas y contiene el nombre del servicio y los espacios de nombres (*namespaces*) utilizables en el documento.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://www.roguewave.com/soapworx/examples/DayOfWeek.wsdl"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" name="DayOfWeek"
targetNamespace="http://www.roguewave.com/soapworx/examples/DayOfWeek.wsdl">

…

</definitions>
```

types

Representa los tipos de datos utilizados como parámetros de entrada/salida por el WebService en cuestión. Mediante el uso de ésta etiqueta se pueden definir tipos

compuestos o más complejos que los existentes. Existen un conjunto de tipos de datos simples predefinidos en los *namespaces* referenciados en la etiqueta <*definitions*>.

```
<types>
        <schema targetNamespace="http://example.com/stockquote.xsd"
xmlns="http://www.w3.org/2000/10/XMLSchema">
          <element name="TradePriceRequest">
               <complexType>
                    <all>
                         <element name="tickerSymbol" type="string"/>
                    </all>
               </complexType>
          </element>
          <element name="TradePrice">
               <complexType>
                    <all>
                         <element name="price" type="float"/>
                    </all>
               </complexType>
          </element>
     </schema>
</types>
```

message

Define un mensaje de entrada/salida intercambiado por un servicio y los tipos de datos que lo componen. Cada etiqueta <*message*> puede contener 0 o más subetiquetas <*part*>, que definen respectivos parámetros de la llamada o respuesta.

```
<message name="DayOfWeekInput">
     <part name="date" type="xsd:date" />
</message>
<message name="DayOfWeekResponse">
     <part name="dayOfWeek" type="xsd:string" />
</message>
```

portType

Contiene un conjunto de operaciones invocables. Para ello combina distintos etiquetas <*message*> en subetiquetas <*operation*>. La etiqueta <*portType*> se puede comparar con una biblioteca de funciones en un lenguaje de programación tradicional.

```
<portType name="DayOfWeekPortType">
     <operation name="GetDayOfWeek">
          <input message="tns:DayOfWeekInput" />
          <output message="tns:DayOfWeekResponse" />
     </operation>
</portType>
```

binding

Refleja la relación entre un <*portType*> y el protocolo mediante el cual se le accede. Se pueden especificar múltiples <*binding*> para un mismo <*portType*>. En la especificación WSDL 1.1 se incluyen una serie de extensiones para poder especificar más directamente ciertos detalles SOAP. Éstas son: <*soap:binding*>, <*soap:operation*> y <*soap:body*>.

```
<binding name="DayOfWeekBinding" type="tns:DayOfWeekPortType">
<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
```

```
<operation name="GetDayOfWeek">
        <soap:operation soapAction="getdayofweek" />
     <input>
          <soap:body use="encoded" namespace="http://www.roguewave.com/soapworx/examples"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
     </input>
     <output>
          <soap:body use="encoded" namespace="http://www.roguewave.com/soapworx/examples"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
     </output>
</operation>
</binding>
```

service

Representa la unión de un <*binding*> y una subetiqueta <*port*>, que es una descripción de la ubicación del servicio y la forma de accederlo.

```
<service name="DayOfWeekService">
     <documentation>Returns the day-of-week name for a date</documentation>
     <port name="DayOfWeekPort" binding="tns:DayOfWeekBinding">
          <soap:address location="http://localhost:8090/dayweek/DayOfWeek" />
     </port>
</service>
```

Un documento WSDL puede contener también otras etiquetas distintas a las anteriormente expuestas. Para una información más detallada al respecto, se recomienda la lectura de [W3O2] o [MSDN1].

4.2.1.4 UDDI (Universal Description Discovery and Integration)

Hasta este punto hemos visto la forma de definir WebServices y los mensajes de entrada/salida que manejan. Pero no la manera en la que pueden ser descubiertos. UDDI [WIKI05] es un estándar (basado en XML) cuyo propósito es el de describir, publicar y encontrar WebServices. Es un framework abierto que puede interoperar a través de SOAP, CORBA y RMI para componer un registro distribuido de servicios.

Una compañía puede publicar en un registro UDDI tres tipos de información:

- White pages: contiene los datos básicos y de contacto sobre la compañía y un identificador único que permita a otros descubrir los servicios puestos a disposición por dicha compañía.
- Yellow pages: contiene más detalles sobre la empresa, como pueden ser categorías de catalogación industrial tradicionales, ubicación geográfica, etc. mediante el uso de códigos estandarizados (SIC, NAICS, UNSPSC, etc.)
- Green pages: en ellas está contenida la información relativa a las especificaciones de los servicios y la forma de accederlo.

Éste es un tema muy extenso, complejo y de escaso interés para el presente documento. Por lo tanto se anima a cualquier lector interesado en profundizar a recurrir a la lectura de [OAS1].

4.2.1.5 Otros estándares de interés

Dado que SOAP se utiliza eminentemente en entornos empresariales, a lo largo de los años han ido surgiendo una serie de estándares[1] adicionales cuyo objetivo es solventar las carencias iniciales de SOAP en dichos entornos.

WS-Security

Es una extensión de SOAP (concretamente del uso de soap:Header) que especifica cómo se puede conseguir autenticación, integridad y confidencialidad en el intercambio de mensajes entre dos entidades SOAP. Esto se lleva a cabo mediante el uso de varios tipos de tokens de seguridad como pueden ser SAML, Kerberos o X.509. Su principal foco de actuación es el uso de XML Signature (firmado digital) y XML Encryption (cifrado) para proveer seguridad extremo a extremo.

WS-ReliableMessaging

En él se describe un protocolo que permite gestionar el intercambio fiable de información entre dos entidades SOAP ante situaciones de fallo de red.

WS-AtomicTransaction

Emula las propiedades de una transacción ACID (Atomicity, Consistency, Isolation, Durability) y, por lo tanto, recoge el concepto de transacción atómica en entornos SOAP.

WS-Coordination

Este estándar tiene como objetivo la creación y propagación de un contexto de ejecución que permita la coordinación de distintos flujos o servicios entre sí.

WS-CDL

Define los mecanismos necesarios para coordinar la ejecución de distintos servicios de forma paralela.

WS-BPEL

Se apoya en los dos anteriores para definir un proceso o flujo de negocio como una composición de servicios distribuidos, sus parámetros de orquestación y de gestión de eventos y excepciones.

[1] Todos los estándares mencionados en esta sección son mantenidos unos por la Organization for the Advancement of Structured Information Standards (OASIS) y otros por el World Wide Web Consortium. En caso de querer profundizar en el conocimiento de dichos estándares, están disponibles para consulta en sus respectivas páginas (www.oasis-open.org y www.w3.org).

4.2.1.6 En resumen

Diseñado desde el principio con el foco puesto en el sector empresarial, SOAP incorpora múltiples estándares para su correcta integración en este tipo de entornos. Algunos ejemplos de ello son la orquestación y composición de servicios, el concepto de estado, el soporte de transacciones ACID, fiabilidad en el intercambio de información o autenticación, integridad y confidencialidad en los mensajes. Sin embargo, SOAP en su conjunto, es un protocolo complejo de implementar y mantener, cuyo medio de expresión es XML y que, por tanto, consume mucho ancho de banda.

4.2.2 Arquitectura REST (REpresentational State Transfer)

El concepto de arquitectura REST, surgió de una necesidad de estandarizar el acceso a recursos en entornos web. Originalmente el término REST se refería a un conjunto de principios de arquitectura [WIKI06]. Sin embargo, en la actualidad se usa en un sentido más amplio para describir cualquier interfaz entre sistemas que utilice directamente HTTP para obtener datos o ejecutar operaciones al más puro estilo RPC. Utilizando para ello cualquier formato de datos (XML, JSON, etc.). Adicionalmente, carece de las abstracciones y sobrecarga de muchos de los protocolos de intercambio de mensajes tradicionales, como por ejemplo SOAP. Estos dos usos diferentes del término REST (obtener datos o ejecutar operaciones) causan cierta confusión, y es sencillo encontrar definiciones ambiguas y diferentes sobre el concepto REST.

Los principios del diseño REST son los siguientes:

- La unidad fundamental de la arquitectura son los recursos.
- Sobre los recursos se puede aplicar un conjunto de operaciones bien definidas como son: PUT, GET, POST, DELETE y OPTIONS.
- Es necesario especificar una definición del contrato de servicio o API.
- Es una arquitectura cliente/servidor. Con ello se consigue separación de conceptos (SoC, separation of concepts) lo que deriva en la independencia en el diseño, desarrollo y mantenimiento de cada una de las piezas, siempre y cuando la definición del contrato de servicio (API) se mantenga inalterada.
- Las operaciones deben ser sin estado. Es decir, el servidor no debe almacenar ninguna información relativa a un contexto de cliente. Dicha información debe permanecer en el cliente y ser enviada, en la medida que sea necesaria, al servidor en cada llamada.

- Las respuestas del servidor deben poder ser cacheadas durante un tiempo definido en la propia respuesta para minimizar las invocaciones al servicio.
- Los servicios deben definirse mediante interfaces uniformes. Para ello se utilizan URIs y éstas deben permanecer constantes para un recurso determinado.

Aquellos servicios que cumplen con los principios de diseño REST son denominados servicios RESTFul o RESTFul WebService por el inherente uso de HTTP. Un ejemplo del uso de operaciones sobre recursos podría ser el conjunto mostrado en la Tabla 1.

Operación	URI	Descripción
GET	/users	Obtener una lista de usuarios
GET	/users/user1	Obtener los datos del usuario user1
PUT	/users/user2	Insertar el usuario user2
POST	/users/user2	Actualizar el usuario user2
DELETE	/users/user1	Eliminar el usuario user1
OPTIONS	/users	Obtener un lista de operaciones sobre un usuario

Tabla 1 - Ejemplo de operaciones REST.

4.2.2.1 JSON (JavaScript Object Notation)

JSON [WIK07] es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML. Se utiliza para el intercambio de datos La simplicidad de JSON ha dado lugar a la generalización de su uso como alternativa a XML, especialmente en entornos web. Si bien es frecuente ver JSON posicionado contra XML, también es frecuente el uso de JSON y XML en la misma aplicación. Por ejemplo, una aplicación de cliente que integra datos de Google Maps con datos meteorológicos en SOAP hace necesario soportar ambos formatos.

La sintaxis de JSON se puede resumir en cuatro puntos:

- Los datos se representan en pares nombre/valor, separados por el carácter '*:*' de la forma *nombre:valor*.
- Las sucesiones de pares se separan por el carácter *','* de la forma *nombre:valor, nombre:valor*.
- Las colecciones de pares o arrays se encierran entre llaves *{}*.
- Los objetos o datos compuestos son también pares pero su valor se encierra entre corchetes *[]*.

JSON vs. XML

Asimismo, el uso de una codificación de tipo ligero como JSON aporta ciertas ventajas frente a XML [JSON1]. Estas son:

- Provee un lenguaje con mayor independencia respecto de la representación de las estructuras de datos.
- Una especificación más simple.
- La cantidad de datos transmitidos y el tiempo de procesamiento es menor y requiere menos recursos.

4.2.2.2 WADL (Web Application Description Language)

WADL es un formato de archivo en XML cuya utilidad es la de describir un servicio RESTFul. WADL representa para los servicios RESTFul lo que WSDL para los WebServices SOA [WIK08]. Al igual que éste último tiene una serie de elementos definidos en su estructura [WADL1].

application

Es el elemento raíz de un documento WADL y especifica los espacios de nombres y esquemas utilizados en la declaración del servicio.

```
<application xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://wadl.dev.java.net/2009/02 wadl.xsd"
  xmlns:tns="urn:yahoo:yn"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:yn="urn:yahoo:yn"
  xmlns:ya="urn:yahoo:api"
  xmlns="http://wadl.dev.java.net/2009/02">

…

</application>
```

grammars

Actúa como contenedor de las definiciones de tipos de datos empleados como entrada/salida de los servicios. Normalmente son referencias a esquemas XML que, a su vez, contienen la descripción de dichos elementos y sus tipos.

```
<grammars>
     <include href="NewsSearchResponse.xsd"/>
     <include href="Error.xsd"/>
</grammars>
```

resources

Es una de las etiquetas fundamentales de WADL. En él se recogen la ubicación y nombre de un grupo de servicios. Son anidables, por lo que elementos *resource* dentro de otros extienden la ubicación (el path de la URI) del recurso inmediatamente superior en la jerarquía.

```
<resources base="http://api.search.yahoo.com/NewsSearchService/V1/">
     <resource path="newsSearch">

…
```

```
    </resource>
</resources>
```

method

Este elemento va siempre contenido dentro de un *resource*. En él se especifica el nombre del servicio o recurso y la operación para accederlo.

```
<method name="GET" id="search">

…

</method>
```

request

Define los nombres, valores y tipos de los parámetros de acceso al *method*. Dichas definiciones se representan mediante el uso de subelementos *param*.

```
<request>
     <param name="appid" type="xsd:string" style="query" required="true"/>
     <param name="query" type="xsd:string" style="query" required="true"/>
     <param name="type" style="query" default="all">
          <option value="all"/>
          <option value="any"/>
          <option value="phrase"/>
     </param>
     <param name="results" style="query" type="xsd:int" default="10"/>
     <param name="start" style="query" type="xsd:int" default="1"/>
     <param name="sort" style="query" default="rank">
          <option value="rank"/>
          <option value="date"/>
     </param>
     <param name="language" style="query" type="xsd:string"/>
</request>
```

response

El elemento *response* representa las distintas opciones de respuesta en función del código HTTP obtenido. Pueden definirse tantos como código hay. El tipo de objeto devuelto se expresa mediante subetiquetas *representation* que describen su tipo de dato y el *mediaType* en que va codificado.

```
<response status="200">
     <representation mediaType="application/xml" element="yn:ResultSet"/>
</response>
<response status="400">
     <representation mediaType="application/xml" element="ya:Error"/>
</response>
```

Un documento WADL puede contener también otras etiquetas distintas a las anteriormente expuestas. Para una información más detallada al respecto, se recomienda la lectura de [WADL1]. Los anteriores bloques WADL componen la descripción de un servicio de búsqueda de Yahoo.

4.2.2.3 Seguridad en RESTFul WebServices

Como hemos visto con anterioridad, la arquitectura REST consiste en un conjunto de prácticas no en un estándar al uso. Asimismo no contempla ningún mecanismo de seguridad. Para ello, se ha de recurrir a protocolos de autenticación estándar o implementar mecanismos de cifrado de datos de forma independiente. Sin embargo, éste último punto evita que los servicios sean utilizables desde una entidad externa que desconozca estos mecanismos y añade complejidad a la gestión de contraseñas y palabras clave. A continuación, se hará una breve exposición sobre algunos conceptos y protocolos utilizados en la securización de servicios RESTFul.

Autenticación vs. Autorización

Son dos conceptos que a menudo se confunden uno con otro. El proceso de autenticación [WIK09] tiene como objetivo confirmar la veracidad de una afirmación. En el escenario actual de arquitecturas de servicios se utiliza para identificar de forma segura y unívoca a uno o ambos extremos de la comunicación.

El concepto de autorización [WIKI10] extiende al de autenticación en tanto en cuanto tiene como objetivo adicional confirmar que el peticionario es quién dice ser y, además, que tiene permiso para acceder al recurso o servicio solicitado.

HTTPS

El mecanismo más sencillo y ampliamente utilizado es el de utilizar Hypertext Transfer Protocol over TLS (HTTPS[2]). Este protocolo [WIKI11] es una variante de HTTP en el que las comunicaciones van cifradas extremo a extremo. Otra particularidad de dicho protocolo es la de permitir la autenticación del servicio ante el cliente.

Autenticación básica HTTP sobre TLS

HTTP provee ciertos mecanismos de autenticación. El más sencillo de ellos se denomina Basic Authetication [WIKI12]. Consiste sencillamente en enviar el usuario y contraseña, codificados en Base64, como parte de las cabeceras. Está soportado por prácticamente la totalidad de los estándares y protocolos actuales. Sin embargo, el par usuario/contraseña viaja en claro en el mensaje HTTP por lo que es susceptible de ser leído por cualquier. Para evitarlo es necesario el uso de HTTPS.

SAML (Security Assertion Markup Language)

Es un estándar basado en XML que permite a un usuario iniciar sesión en distintos servicios afiliados entre sí. SAML especifica cuatro componentes [WIKI13]:

[2] También se suele denominar HTTP Secure o HTTP sobre SSL por ser éste el predecesor de TSL.

- Afirmaciones: este componente contiene la información relativa a la identidad del solicitante y sus permisos.
- Protocolos: especifica cómo se empaquetan los distintos componentes SAML en las peticiones y respuestas.
- Enlaces: detalla la forma de incluir un mensaje SAML en otro protocolo de mensajería, como puede ser HTTP, SOAP, etc.
- Perfiles: describe la forma en que se combinan los anteriores componentes para un determinado caso de uso. El más importante de ellos es Web Browser Single Sign-On Profile, que permite el acceso a distintos servicios una vez autenticado en uno de ellos.

OAuth 1.0a (Open standard for AUTHorization)

Es el más seguro de los protocolos explicados. Es un protocolo de autorización basado en firma digital de uso muy extendido en la industria [WIKI14]. Ésta versión de OAuth utiliza una firma criptográfica (normalmente HMAC-SHA1) que se combina con un token secreto y otra información. Su gran ventaja es que nunca se transmite directamente la contraseña. Lo que elimina la posibilidad de ser leída durante la transmisión. Por ese mismo motivo, no impone el uso de una comunicación cifrada por TSL o similar. Sin embargo, éste esquema de seguridad conlleva una alta complejidad y coste computacional derivado del procesamiento criptográfico implícito.

OAuth 2

Pese a que por su denominación induce a pensar que es una evolución del anterior, no es así. En este caso AUTH significa autenticación. Éste protocolo elimina el uso de firmas digitales por lo que no se requieren cálculos criptográficos. Para dotar de seguridad al sistema se delega en el cifrado con TSL. Sin embargo, se han detectado algunos fallos de seguridad y rendimiento. Por ese motivo, se recomienda OAuth1.0a en entornos donde la información transmitida es sensible [OAUTH1].

4.2.2.4 En resumen

REST, al contrario que SOAP, ha surgido y evolucionado dentro de la comunidad de desarrolladores web, y lejos de procesos de estandarización, hasta convertirse en un estándar de facto en el sector. Sus beneficios son que es sencillo y, por tanto, fácil de implementar y mantener. Acepta múltiples formatos de datos, como JSON, lo que, a su vez, aporta ligereza en las comunicaciones. También impone una separación en el diseño entre las implementaciones del cliente y del servidor. Además, los clientes pueden cachear la información recibida previamente. Por último, REST presenta mejor rendimiento y escalabilidad que SOAP [SvR1]. Sin embargo, sólo funciona sobre HTTP y no provee ningún mecanismo propio de seguridad o transaccionalidad.

4.2.3 Conclusiones

Como hemos visto, SOAP se integra mejor con los procesos empresariales y sus requerimientos; sin embargo, lo hace a un alto coste. REST, por otro lado, es una arquitectura más dinámica y con un extendido uso en entornos web; pero carece en su definición de los mecanismos de seguridad, fiabilidad y orquestación necesarios para ciertos procesos.

Actualmente la mayor parte de grandes figuras del mundo tecnológico utilizan eminentemente REST tanto a nivel externo como interno [SvR2]. A priori, pues, la mejor opción puede parecer la de implementar una arquitectura REST. Sin embargo, la naturaleza intrínseca de la actividad de las empresas del sector financiero, obliga a tomar ciertas precauciones en el diseño de la arquitectura.

En base a todo lo anterior, el diseño final de la arquitectura de servicios empleará una solución híbrida SOAP y REST. La primera se empleará en los procesos internos de la entidad y todos aquellos externos que requieran de un añadido de seguridad. REST será utilizado para todo lo demás, especialmente en el EdT y los portales de acceso de clientes. El uso de REST no implica rebajar los requerimientos de seguridad en el acceso. Por lo que será necesario proveer los mecanismos de autenticación, integridad y confidencialidad suficientes mediante el uso de comunicaciones cifradas, OAuth, etc.

Esta solución de compromiso permite aunar el dinamismo, la versatilidad y la ligereza de REST con la seguridad, transaccionalidad y el concepto de contrato y estado de SOAP.

4.3 Servicios en la nube

Durante décadas, hubo ciertas aproximaciones al concepto que hoy en día conocemos como Cloud Computing [WIKI15], o servicios en la nube. Básicamente, el concepto detrás de Cloud Computing se refiere principalmente a productos y servicios accesibles a través de una red de comunicaciones, normalmente Internet. Esta definición es un tanto vaga y no dista mucho de la de un servicio, de cualquier naturaleza, accesible a través de una red. Lo que se lleva haciendo desde la invención de Internet en los años 50. Veamos por qué la computación en la nube es tan especial.

A principios de siglo, Amazon se dio cuenta que disponían de una infraestructura sobredimensionada para hacer frente a los picos de las fechas más comerciales y cubrir los distintos husos horarios. Ese sobredimensionamiento derivaba en un exceso de producción de TIC. En ese punto crearon AWS (Amazon Web Services) [pAWS01] con el objetivo de diversificar el modelo de negocio de Amazon y vender dicho excedente a terceros de forma profesional. Precisamente éste es el concepto fundamental del Cloud Computing: pensar

en las capacidades de TIC como un producto de mercado que se puede producir, cuantificar, dimensionar, etc. La clave consiste en entender los servicios TIC como una materia prima o una commodity.

Sobra decir que ese nuevo modelo de negocio tuvo mucho éxito y derivó en la creación de un nicho de mercado. Llegados a este punto, se crearon multitud de empresas enfocadas a satisfacer una creciente demanda. En aproximadamente 10 años, el sector del Cloud Computing se ha establecido como uno de los más boyantes del sector TIC y sigue en constante crecimiento. Tanto es así que se estiman unos ingresos de 106 mil millones de dólares para 2016 con un crecimiento aproximado el 21% respecto a 2015 [FORB1], sólo en el apartado Software as a Service (SaaS).

El éxito del Cloud Computing radica en proveer un conjunto de recursos TIC, como pueden ser redes, servidores, almacenamiento, aplicaciones y servicios, y hacerlo de forma que puedan estar disponibles para cualquier consumidor, independientemente de su nivel de conocimientos, en un tiempo reducido y con un esfuerzo mínimo. En términos prácticos, el consumidor ya no compra un servicio incurriendo en todos los gastos derivados, si no que lo alquila. Los servicios en la nube se pueden ofrecer en base a modelos de servicio (infraestructura, plataforma y aplicación) y de despliegue (nube pública, privada, híbrida, etc.).

4.3.1 Características esenciales

Las tecnologías de computación en la nube presentan una serie de características que las hace especialmente interesantes para el caso que nos atañe [WIKI15].

Auto-aprovisionamiento bajo demanda

El consumidor puede configurar una serie de reglas en base, normalmente, a parámetros de rendimiento. En función de dichas condiciones, la nube puede provisionar o redimensionar un conjunto de servicios o recursos (como la capacidad de un servidor, el tamaño de un almacenamiento en red o el número de instancias de un servicio) de forma autónoma.

Compartición de recursos

Los recursos del proveedor están agrupados y se comparten para atender a varios consumidores a la vez, según un modelo llamado multi-tenant [WIKI16]. Según éste modelo, varios recursos físicos y virtuales son asignados dinámicamente de acuerdo a las necesidades de cada consumidor. Esta opacidad respecto al origen real de los recursos hace que el cliente no tenga control o conocimiento sobre la ubicación exacta de dichos recursos. Se puede, en ocasiones, especificar una ubicación de forma abstracta como un país, una región o un determinado centro de datos.

Escalabilidad y elasticidad

Los recursos requeridos por un servicio pueden ser asignados y liberados rápidamente, en muchos casos de forma automática, para adaptarse a la demanda. Para el consumidor del servicio, los recursos disponibles se muestran, a menudo, como ilimitados y se pueden adecuar a las necesidades en cualquier momento y cantidad.

Cuantificación

Los sistemas basados en Cloud Computing controlan y optimizan el uso de recursos de manera automática utilizando unos patrones de medida apropiados al tipo de servicio ofrecido como tiempo de cómputo, ancho de banda, almacenamiento, número de transacciones, cuentas de usuario activas. Así, es posible visualizar, controlar y confeccionar informes sobre los recursos consumidos proporcionando transparencia tanto para el proveedor como para el consumidor del servicio.

Coste

En el modelo de negocio de Cloud Computing se factura por uso de recursos. Esto permite, por un lado, reducir las barreras de entrada de nuevas empresas que se apoyen en el uso de éstos servicios. Y por otro, transformar las inversiones indirectas en costes directos de operación.

Accesibilidad

La naturaleza deslocalizada y descentralizada (respecto del cliente) de este tipo de servicios facilita la accesibilidad tanto para el consumidor final como para las labores de mantenimiento y control.

4.3.2 Modelos de servicio

Toda aplicación multicapa, por sencilla que ésta sea, involucra el uso de multitud de componentes de distinta índole como pueden ser los dispositivos de red, servidores con características diferentes en función de su finalidad, un sistema operativo, middleware de bases de datos, sistemas de autenticación, colas de mensajes y, finalmente, una o varias aplicaciones actuado de forma aislada o cooperativa. Son, por lo tanto, ecosistemas complejos que requieren de conocimientos en distintos campos.

Las empresas de Cloud Computing proveen distintos modelos de servicio [NIST1] en función del nivel de abstracción que requiera el cliente. Así, se pueden segmentar los servicios ofrecidos como de infraestructura, de plataforma o de aplicación. La Ilustración 7 muestra un pequeño esquema de cómo se relacionan y dónde se ubican las capas y tecnologías mencionadas. Además, la mayoría de las soluciones de Cloud Computing, se construyen sobre tecnologías de virtualización. Lo que posibilita el cumplimiento de las características expuestas en la sección anterior.

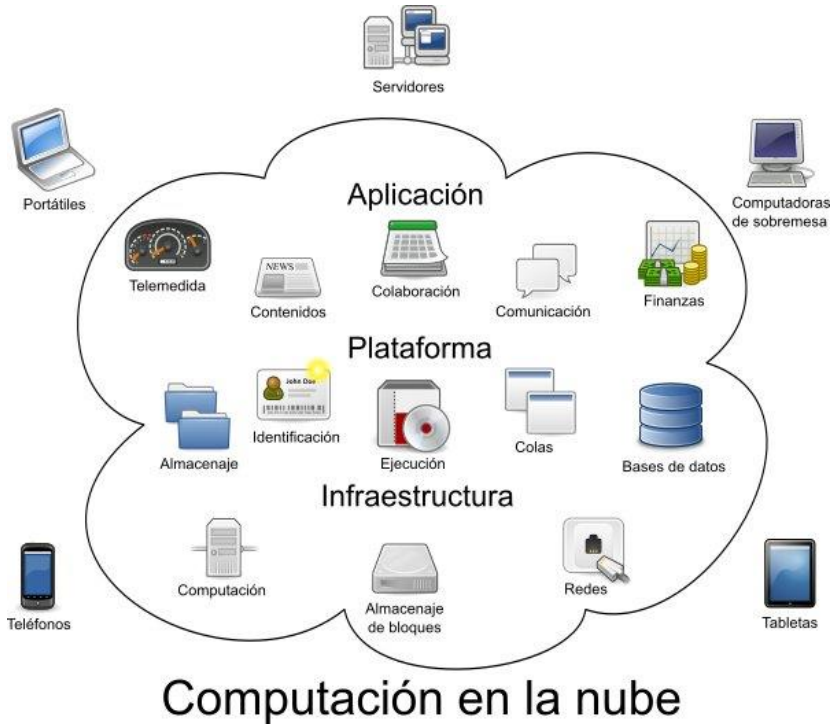

Ilustración 7 - Modelos de servicio del Cloud Computing [WIKI17].

Software as a Service (SaaS)

Bajo este modelo, el cliente consume de forma remota las aplicaciones que un proveedor pone a su disposición mediante una infraestructura propia, o no, de tipo nube. En cierto modo, SaaS [WIKI18] es muy similar al modelo de provisionamiento de software de thin client, donde los clientes, por lo general los navegadores web, proporcionan el punto de acceso al software que se ejecuta en la nube. Este modelo de servicio es la forma más familiar para los consumidores de servicio en la nube.

SaaS, por tanto, traslada al proveedor las tareas de administrar la infraestructura subyacente que incluye equipos de red, servidores, sistemas operativos, almacenamiento, middleware, etc. El consumidor tan sólo puede gestionar algunos aspectos relacionados con la configuración de las aplicaciones y los usuarios. Entre las aplicaciones SaaS más conocidas estarían Office 360 y Google Apps (ofimática), Dropbox (almacenamiento), Spotity (música), Youtube (videos), Salesforce (CRM), GoToMeeting (videoconferencia), QuickBooks (contabilidad), Basecamp (gestión de proyectos), etc.

El uso de aplicaciones SaaS tiende a reducir el coste de mantenimiento de una aplicación, eliminando la necesidad de personal técnico cualificado dedicado a gestionar instalar, administrar y actualizar el software. Además, reduce el coste de licencias del software a corto plazo pues las aplicaciones SaaS son generalmente comercializadas en un modelo de suscripción. Por último, dicho modelo de tarificación garantiza disponer de la última versión del software sin incurrir en costes adicionales.

Platform as a Service (PaaS)

PaaS [WIKI19] funciona a un nivel más bajo que SaaS. Típicamente provee una plataforma sobre la que el consumidor puede desarrollar y desplegar aplicaciones propias o de terceros. Los proveedores de PaaS abstraen al cliente de gran parte del trabajo de gestionar los servidores y dan al consumidor un entorno en que el software y sistema operativo de servidor, así como el propio hardware y la infraestructura de red subyacente, están controlados. El cliente puede entonces centrarse en la parte comercial, la escalabilidad o el desarrollo su propio producto o servicio. Sin embargo, las aplicaciones deben ser desarrolladas utilizando lenguajes, bibliotecas, servicios y herramientas soportados por la plataforma. Un ejemplo de la integración y flexibilidad posible en entornos nube sería Dropbox que es un Saas construido sobre un Paas de otros proveedores, principalmente Amazon.

Algunos ejemplos de PaaS incluyen Heroku, Google App Engine, AWS Elastic Beanstalk, Microsoft Azure o Pivotal.

Infrastructure as a Service (IaaS)

IaaS [WIKI20] es el escalón inferior en la pila de abstracción de un sistema en la nube. Una plataforma IaaS se compone de tres recursos básicos: almacenamiento, máquinas y capacidad de red. Estos recursos son todos virtuales y se proveen de manera opaca, automatizada y cuantificada a partir de una capa física real a la que el consumidor nunca tiene acceso directo. El proveedor del servicio agrega capacidad (nuevas líneas y elementos de red, sistemas de almacenamiento y servidores físicos) a la nube. Ésta los integra de forma transparente para dar servicio a los clientes que la contratan en cantidades determinadas.

El consumidor no controla o administra la infraestructura subyacente, pero sí tiene control sobre sistemas operativos, almacenamiento, aplicaciones desplegadas, y posiblemente control limitado sobre algunos aspectos de la red de comunicaciones como firewalls o cortafuegos. Asimismo, tiene la posibilidad de escalar su sistema mediante auto-aprovisionamiento con unos cuantos clicks; sin tener que invertir en la planificación de capacidad o el mantenimiento físico y la gestión de la misma.

Algunos ejemplos de empresas y productos IaaS son Amazon EC2, Rackspace, GoGrid, Terremark, IBM SoftLayer, CSC CloudCompute, etc. En España dos buenos referentes son Arsys y Acens.

4.3.3 Modelos de despliegue

Además de los distintos modelos de servicio, también se definen [NIST1] cuatro modelos de despliegue sobre la infraestructura física.

Nube privada

La infraestructura al completo es explotada en exclusiva para una sola organización. Puede ser propiedad de la organización y estar dirigida y operada por ésta, por un tercero, o por alguna combinación de ambas y puede residir físicamente dentro o fuera de las instalaciones de la organización. Los consumidores de las nubes privadas son, normalmente, los diferentes departamentos de una empresa, cada uno con sus requisitos y niveles de servicio propios e independientes.

Nube pública

La infraestructura es proporcionada para uso del público en general. Puede ser propiedad de una compañía, una institución educativa o gubernamental y estar dirigida y operada por éstas, por un tercero, o por alguna combinación de ambas y reside físicamente en las instalaciones del proveedor. En la mayoría de ellas, se establece un régimen de pago por uso.

Nube híbrida

La infraestructura es una composición de dos o más nubes que están separadas pero que se combinan mediante tecnologías estándar o propietarias permitiendo portabilidad de datos y aplicaciones. Los escenarios típicos de éste tipo de nubes son, por un lado el cloud bursting o balanceo la carga entre infraestructuras para que una nube pública soporte el excedente de demanda de una nube privada. Por otro, en ciertas situaciones existen una serie de requisitos de confidencialidad por los cuales ciertos datos o procesos se soportan en una nube privada y el resto en una pública.

Community cloud

La infraestructura es proporcionada para uso exclusivo de un grupo específico de consumidores cuyas organizaciones comparten intereses u objetivos similares. Puede ser propiedad de una o varias de las organizaciones y estar dirigida y operada por éstas, por un tercero, o por alguna combinación de ambas y puede residir físicamente dentro o fuera de las instalaciones de la organización. De esta forma, se comparten los costes de entrada y explotación de una nube a la par que se disfruta de la mayoría de beneficios de una nube privada.

Otros tipos

Pese a su juventud, las tecnologías cloud están evolucionando rápidamente. Algunas de estas innovaciones van dirigidas a modificar los modelos de despliegue actuales [UoM1]:

- Nube distribuida [WIKI21]: en este modelo de despliegue la infraestructura que soporta la nube se compone de grupos de máquinas distribuidos en distintas localizaciones pero conectadas a una misma red o servicio de concentración de comunicaciones. Un subtipo serían las nubes ad-hoc o peer-to-peer en las que los

recursos son dispuestos por voluntarios alrededor del mundo. La mayor complejidad en este tipo de nubes es gestionar la volatilidad de los recursos.

- Intercloud [WIKI22]: es la evolución conceptual del Cloud Computing. Este concepto se puede resumir como una nube de nubes. Una agregación de nubes públicas interconectadas que se sirven unas a otras bajo una serie de requisitos determinados.
- Multicloud [WIKI23]: es el concepto más heterogéneo de todos los expuestos, una multicloud se refiere a un concepto de arquitectura en la los servicios se obtienen a distintos niveles (SaaS, PaaS e IaaS) de distintos tipos de nube. Estos esquemas están aumentando en relevancia pues permiten a un consumidor independizarse respecto de los medios y condiciones dispuestos por un proveedor de servicios en la nube.

4.3.4 Sistema de tarifas

Hablar de precios de productos SaaS es muy ambiguo. Al igual que cada fabricante fija el precio de una licencia en el modelo tradicional, en el escenario de aplicaciones en la nube ocurre algo parecido. Sin embargo, los esquemas de tarificación para venta al público suelen seguir una estrategia que a menudo se denomina Freemium [WIKI25]. En este modelo existe un nivel de servicio de uso gratuito pero limitado en tiempo o funcionalidad. Si el cliente está interesado en el producto tiene la opción de subir a uno o más niveles de servicio, en los que dispone de mayores capacidades, mediante el pago de una subscripción periódica. Este esquema permite aumentar la base de clientes potenciales, así como la viralidad del producto mediante un sistema de marketing delegado.

En el ámbito de productos IaaS los esquemas de precios se vuelven más homogéneos, a la par que complejos. Hablar de tarifas es una labor muy complicada pues depende mucho de las necesidades de cada cliente. Al igual que los productos SaaS, es bastante común que los proveedores ofrezcan una capa básica de forma gratuita. La tarificación a partir de esa nivel gratuito se factura por uso y volumen.

Por último, en los sistemas de precios de los servicios PaaS se puede encontrar tanto el modelo de subscripción como el esquema de pago por uso.

4.3.5 Seguridad, privacidad y propiedad intelectual

Con el Cloud Computing, y su reciente auge, surgen una serie de preocupaciones [WIKI24]. La raíz del problema reside en la propia naturaleza de la idea de nube. Según este concepto, la información y procesos de un consumidor residen y se ejecuten en las

instalaciones de un tercero en un lugar determinado por él y regido por unas leyes distintas a las del consumidor del servicio. El consumidor, por tanto, pierde el control y la propiedad de los mismos. Esto da lugar a ciertos aspectos preocupantes a día de hoy.

Privacidad

En lo referente a la gente de a pie, la situación no es muy halagüeña. Actualmente existen innumerables servicios en la nube ofrecidos de forma gratuita o de pago y que se utilizan diariamente como pueden ser Facebook, Twitter, Google Apps, Outlook, Evernote, Dropbox, Flickr, etc. Muchos de estos servicios gratuitos lo son debido a que explotan dicha información, normalmente para ganar mayor conocimiento del individuo con el objetivo de obtener beneficios directos o indirectos a partir de ella.

En un escenario empresarial hay que añadir la posibilidad de que su información sea objeto de espionaje industrial. Estas preocupaciones están forzando a que las organizaciones con capacidades suficientes construyan nubes privadas o híbridas para minimizar el impacto de la exposición de información. Sin embargo, organizaciones más pequeñas no disponen de recursos suficientes y se pueden ver afectados por esta falta de confidencialidad.

Distintos países tienen distintos marcos regulatorios que confieren mayor o menor confidencialidad a la identidad y los datos tratados en los entornos de Cloud Computing. La solución definitiva consistiría en un marco regulatorio común para todas las actividades derivadas del uso de las tecnologías Cloud. Sin embargo, el número de intervinientes, tanto empresariales como gubernamentales, que estarían implicados y la disparidad de intereses de unos y otros hace inviable, al menos en la práctica, dicha solución.

Una medida de mitigación que puede aplicar un consumidor sea cual sea su naturaleza, consistiría en recurrir a servicios que garanticen contractualmente la protección de los datos o que, al menos, garanticen regirse por un marco regulatorio estricto. En caso de no poder asegurar la confidencialidad por la vía regulatoria, un usuario puede, en determinados servicios, optar por el uso de soluciones criptográficas en el uso de su información. Pero esto no es aplicable a todos los servicios y no todos los usuarios tienen los conocimientos necesarios.

Safe Harbour

En relación al punto anterior, la Unión Europea establece un contexto proteccionista para la privacidad de sus ciudadanos mucho mayor que los Estados Unidos. Sin embargo, las empresas más fuertes del sector tecnológico son, en su mayoría, de origen estadounidense. Y por tanto, sus actividades no se rigen bajo las normativas europeas. En los últimos años, organismos de la UE y EEUU han creado una serie de principios internacionales denominados Safe Harbour, cuyo objetivo es el de garantizar que las empresas estadounidenses certificadas cumplan con los siete principios clave recogidos en

la Directiva 95/46/CE [SHC1] [SHC2]. Sin embargo, el compromiso de cumplimiento de dichos principios no siempre es suficiente y ello genera ciertas suspicacias [SHC3].

Propiedad intelectual

En lo relativo a la propiedad intelectual, existen múltiples ejemplos, siendo Facebook el más llamativo de ellos, de la falta de protección que rodea a los servicios en la nube. De acuerdo con los términos y condiciones de algunos servicios, la propiedad intelectual de la información gestionada por el proveedor del servicio no pertenece, a veces ni siquiera en parte, a usuario que la creó. Esto arroja muchas dudas al respecto de los beneficios de explotación de dicha información.

Seguridad

Existe también una creciente preocupación en lo concerniente a los procesos depositados en infraestructuras Cloud de terceros. En este punto podemos hablar de varios de riesgos de seguridad:

- La información queda depositada en la infraestructura de un tercero y el consumidor no puede controlar el acceso directo a la misma por parte de un empleado del proveedor.
- El Cloud Computing se apoya fuertemente en el uso de la virtualización, lo que añade una capa adicional a securizar.
- La compartición de recursos entre distintos consumidores crea un escenario en el que los datos no están aislados unos de otros y es posible que un cliente tenga acceso a la información de otro.
- El acceso de un cliente a su información se realiza, normalmente, a través de Internet. Esta forma de acceso desde el exterior posibilita que un tercero capture información relevante o credenciales de acceso.
- Por un lado, existe una compleja estructura de dependencias entre unos servicios en la nube y otros. Por otro, una infraestructura Cloud puede verse sometida a ataques informáticos. Esta dependencia y vulnerabilidad podría producir un efecto en cadena que dejaría a multitud de clientes sin servicio de forma directa o indirecta.

Actualmente se toman innumerables medidas de control y seguridad en los entornos Cloud. Sin embargo, produce cierto desasosiego y sensación de vulnerabilidad el desconocimiento, por parte del cliente final, de cuáles son dichas medidas.

4.3.6 Integración PaaS

El acceso a los servicios de las plataformas PaaS se publican comúnmente mediante servicios REST. Los proveedores facilitan unas librerías que abstraen al consumidor de gran parte de la gestión y acceso intermedios de dichos servicios. Estas librerías a menudo se ofrecen en varios lenguajes de programación distintos para facilitar la integración con su plataforma. Lo más normal es encontrar dichas librerías en Java, .NET, Ruby, Python y PHP.

El desarrollo y adaptación de una solución a los APIs de un proveedor concreto supone un coste de desarrollo a tener en cuenta, así como un factor de retención a la hora de migrar a otra plataforma distinta.

4.3.7 En resumen

Los beneficios del Cloud Computing son innegables. Sin embargo, la forma en que se gestiona a día de hoy la cuestión de la privacidad es, cuanto menos, preocupante. Y más si tenemos en cuenta la cantidad y el tipo de la información que manejan las entidades bancarias a día de hoy.

La solución idónea pasa por construir una nube híbrida. Bajo este esquema, todos los procesos e información sensible deben quedar confinados en la parte privada de la nube donde la entidad tendrá control total sobre ellos. El resto de procesos e información se podrán ubicar en los servicios PaaS o IaaS de un proveedor externo; siempre y cuando éste almacene los datos en servidores en la UE o que, como mínimo, cumpla con los principios Safe Harbour.

4.4 Arquitectura global

Ya conocemos los dos puntales básicos sobre los que se construirá la nueva arquitectura. Sin embargo, quedan aún muchos más elementos por desgranar. Empecemos por la arquitectura a nivel global.

Como ya se expuso en el capítulo 2, los modelos de despliegue actuales del EdT presentan ciertas carencias que limitan las capacidades de evolución del modelo productivo. Por lo tanto, es necesario plantear una transformación de la arquitectura global que pueda soportar los procesos de negocio presentes y futuros, y que satisfaga los objetivos impuestos. La Ilustración 8 muestra el esquema general de la arquitectura TIC propuesta. A continuación, se desglosarán y explicarán las piezas más importantes.

Nube híbrida corporativa (NHC)

Se compone de un conjunto de servicios SaaS, PaaS e IaaS que compondrán la base de los procesos de negocio. En función de la criticidad o la naturaleza de los datos que se manejen, irán ubicados bien la parte privada bien en la pública. Pero habrá que proveer canales de comunicación suficientes entre ambas partes para que la calidad de servicio no se vea afectada. Existen multitud de proveedores de PaaS e IaaS de referencia en el sector, la elección de cuál de ellos utilizar dependerá del sistema de tarificación, de si tienen o no CPDs en la UE, de las garantías de servicio que ofrezcan y de las facilidades de integración con sus infraestructuras. Sin embargo, cabe resaltar que un movimiento prudente sería el de no contratar servicios solamente a un proveedor.

Ilustración 8 - Nueva arquitectura global

Red de oficinas

Aunque ya se describió previamente el nuevo modelo de oficinas, cabe resaltar que se compone de dos tipos de entidades: los puestos de trabajo y las Islas de Dispositivos. Estos elementos tienen naturalezas y propósitos muy diferentes. Por un lado, los nuevos puestos de trabajo, y la forma en que serán utilizados de cara al cliente, imponen el uso de una red wifi en la oficina, así como la creación de un escritorio de trabajo ligero, atractivo visualmente y con una alta usabilidad.

Las IdDs, por otro lado, son entidades que no deberán requerir interacción humana más allá de su instalación y mantenimiento. Para ello deberán incorporar varias piezas de software fundamentales: un servidor web donde se ubiquen todos los servicios que serán consumidos y unos componentes software que le permitan comunicarse con los servicios de la NHC e integrarse en los flujos de trabajo. En cuanto al hardware, obviamente contendrá los dispositivos y un ordenador, no especialmente potente y de reducidas dimensiones, que vaya ubicado dentro de la IdD y donde se ejecuten los procesos antes mencionados.

Con el objetivo de simplificar la correcta ubicación de los equipos dentro de una oficina determinada, se aplicaran reglas de asignación de direcciones IP segmentadas por rangos. Cada rango corresponderá a una oficina determinada, será para su uso en exclusiva y se registrará en una base de datos en el momento de la configuración de la red. Asimismo, el tamaño de los rangos dependerá de las necesidades y tamaño de cada oficina. De esta forma, bastará aplicar una máscara de red determinada y una pequeña consulta para saber en cada momento qué equipos hay en cada oficina y de dónde provienen las peticiones.

Sistemas e infraestructura de soporte a operaciones

El día a día de una entidad involucra la actividad de numerosas personas en el CPD. Por nombrar algunos, existen los departamentos de gestión de incidencias, mantenimiento de infraestructura, desarrolladores, etc. Con las nuevas propuestas del capítulo previo, además habría que añadir a la anterior lista el departamento de atención al cliente y el de gestores comerciales.

Cada uno de estos departamentos debe disponer de un determinado conjunto de herramientas y acceso a recursos. Todo ello integrado y disponible a través de la NHC.

Sistemas Legacy

Las entidades bancarias tienen un gran histórico de plataformas Legacy, que son sistemas antiguos o propietarios que, por un motivo u otro, deben seguir prestando servicio. Por ello, habrá que disponer los medios necesarios para que puedan ser administrados por los equipos de soporte a operaciones o accedidos desde los nuevos procesos de la NHC.

Sistemas de terceros

Una entidad financiera no existe como un ente aislado de otras organizaciones. Son necesarias las relaciones con otras entidades financieras, consumidores de sus servicios, socios empresariales y proveedores. Todo ello se lleva a cabo mediante conexiones al exterior que deberán configurarse a la medida de las necesidades puntuales de cada caso.

Clientes

Por último, aunque no por ello menos importante, hay que tener en cuenta el uso que los clientes harán, de forma remota, de los servicios ofrecidos. Este punto no es objeto del

presente documento. Sin embargo, cabe recordar los principios expuestos en los capítulos precedentes acerca de la omnicanalidad y la experiencia de usuario uniforme. Habrá, por tanto, que diseñar el portal de acceso y las aplicaciones móviles de forma que el usuario pueda utilizar los servicios de la entidad y tenga una experiencia rica y homogénea. Así mismo, se deberá tener en cuenta que pueden existir canales alternativos como el contacto telefónico o por video conferencia con sus gestores, redes sociales, etc. Todo ello integrado con los servicios en la nube más comunes de uso diario.

4.5 Diseño de la NHC

4.5.1 Elección de proveedores

En lo concerniente a la NHC, existen múltiples proveedores que se podrían ajustar a las necesidades de una entidad. De acuerdo con varias fuentes [GAR1] y [GAR2], el líder indiscutible en servicios Cloud es Amazon, seguido, a cierta distancia, por Microsoft, Google, CenturyLink, VMware, IBM, Rackspace, etc. En lo que a empresas españolas se refiere podríamos mencionar a Arsys y Acens, entre otras. Veamos algunos de ellos.

Las ventajas del primero, Amazon, son que prácticamente creó el sector de la computación en la nube, lleva años siendo líder en cuota de mercado y dispone de centros de datos en la UE, concretamente en Dublín y Frankfurt. En sus infraestructuras se ejecutan procesos de todo tipo y fundamentan la base de muchos otros servicios SaaS de terceros (Dropbox, Expedia, Foursquare, etc.). De hecho, dispone de más potencia de cómputo que los otros 14 líderes del mercado por detrás suyo. Asimismo, dispone del mayor catálogo de productos y de una amplia red de socios que aportan servicios de consultoría específica enfocada en AWS.

Microsoft llegó algo más tarde al mercado de los servicios Cloud. Pero partiendo de un enfoque más empresarial ha logrado acumular las mayores tasas de crecimiento año a año hasta colocarse en segundo lugar. Actualmente dispone del doble de capacidad de cómputo que el resto de competidores por detrás suyo. El hecho de haber llegado tan tarde al sector, produce ciertas inquietudes en cuanto a su capacidad para soportar procesos críticos o la falta de especialistas con experiencia. De hecho, sus cifras de tiempo de indisponibilidad de servicio en el último par de años son ciertamente alarmantes [CCUT1] [CCUT2].

Pese a llevar años ofreciendo servicios SaaS en la nube, Google ha entrado en el mercado PaaS e IaaS recientemente. Sin embargo, la experiencia del gigante es un valor seguro y sus cifras en cuota de mercado no paran de subir.

Tal y como se comentó anteriormente, es conveniente contratar servicios de dos proveedores distintos. El desarrollo, obviamente, será más costoso. Pero con una arquitectura software diseñada convenientemente, esta decisión aportará independencia respecto de proveedores. Así como la posibilidad de migrar servicios de uno a otro proveedor para aprovechar las diferencias en cuestión de tarifas una vez se conozca mejor la naturaleza del consumo en que se incurre. Lo que redundará en beneficios tangibles a medio y largo plazo.

Para la elección de proveedores habrá que tener en cuenta varios factores tanto internos de la entidad en cuestión, como del proveedor. Para la parte pública de la NHC se proponen como proveedores Amazon Web Services [pAWS01] y Google Cloud Platform [pGCP01]. Por varios motivos:

- Ambos tienen la certificación Safe Harbour y disponen de centros de datos en la UE.
- Disponen numerosos centros de datos a escala global. Empresas con actividades fuera de Europa pueden aprovechar este punto para integrar las actividades de sus filiales sin mermar la calidad de servicio.
- Ambas dos tienen unas cifras de fiabilidad muy elevadas[3].
- Adicionalmente, Google dispone de herramientas en la nube de uso común para los empleados como son Gmail, Drive y Docs.

4.5.2 Parte pública

Para una solución tan compleja como la actual será necesario el uso de multitud de servicios. Ambos proveedores elegidos ofrecen un catálogo de productos muy extenso, por lo que detallaremos brevemente los que son más típicos.

Entorno de ejecución web

Como ya se ha establecido, la base unidad mínima de la solución son los servicios web, bien SOAP o REST. Para poder hacer uso de ellos es necesario disponer de un entorno de ejecución PaaS o unos servidores de aplicaciones montados sobre infraestructuras IaaS. Las ventajas de contratar una solución PaaS radican en que toda la gestión del ciclo de vida de los servidores y su dimensionamiento se delega en el proveedor. Google ofrece Google App Engine [pGCP02] y Amazon AWS Elastic Beanstalk [pAWS02].

[3] De acuerdo con los datos obtenidos en www.cloudharmony.com, en el intervalo de un año (08/2014 - 08/2015), el tiempo de indisponibilidad para los servicios de cómputo en la nube ha sido de 1,3 h. en Amazon EC2 y 9,4 h. en Google Compute Engine. Respecto a los servicios de almacenamiento, las mediciones muestran caídas de 3,6 h. en Amazon S3 y 10,1 min. En Google Cloud Storage.

Bases de datos

Por norma general, las aplicaciones necesitan almacenar información de forma más o menos estructurada. Existen para ello numerosas tecnologías y variantes en función de las necesidades concretas de un proceso. Las más habituales son las bases de datos relacionales o SQL, las no relacionales denominadas NoSQL o aquellas diseñadas específicamente para el almacenamiento masivo de datos.

Para satisfacer estas necesidades, Google dispone de Cloud SQL [pGCP03], Cloud Datastore [pGCP04] y Cloud Bigtable [pGCP05] respectivamente. Amazon, por su lado, ofrece Relational Database Service [pAWS03], DynamoDB [pAWS04] y Redshift [pAWS05]. Ambos proveedores disponen además de otros servicios más específicos como almacenamiento de baja latencia, cachés compartidas, etc.

Caché distribuida

En ciertas situaciones puede ser necesario disponer de una caché compartida por un conjunto de servidores, normalmente web. Los usos más comunes son acelerar el acceso a datos y no sobrecargar los servidores de bases de datos. Para estas situaciones, Amazon ofrece ElastiCache [pAWS06] y Google una cache distribuida accesible desde las instancias de App Engine.

Almacenamiento

Las bases de datos almacenan datos de operación. Pero también será necesario contar con sistemas de almacenamiento. Existen muchos productos pensados para tareas concretas como pueden ser la distribución de contenidos, el archivado de documentación, la emulación de sistemas de ficheros, etc. Sin embargo, por ahora bastará con un sistema más tradicional como Amazon Simple Storage Service [pAWS07] o Google Cloud Storage [pGCP06].

Colas de mensajes

Las colas de mensajes simplifican las comunicaciones asíncronas entre entidades de la nube. Uno de los usos más frecuentes es el de comunicar eventos entre distintos procesos distribuidos. Para cubrir esta funcionalidad, Google dispone de Cloud Pub/Sub [pGCP07] y Amazon de Simple Queue Service [pAWS08].

Big Data

Aunque las técnicas de análisis de datos llevan muchos años existiendo y evolucionando, ha sido a raíz de la popularización del Big Data cuando han terminado de despegar. Como se introdujo en el capítulo 3, es necesario que las entidades consigan aumentar el conocimiento que tienen de sus clientes. Y el Big Data es la herramienta perfecta para ello. Para ello Google brinda BigQuery [pGCP08] y, Amazon, Elastic MapReduce [pAWS09].

Virtualización de servidores

También será necesario disponer de ciertos servicios gestionados por la propia entidad, bien porque no los ofrece ningún proveedor, bien porque no lo hace con las características requeridas. Para ello se utilizan máquinas virtuales sobre las que instalar las piezas deseadas. Un ejemplo podrían ser servidores LDAP o directorio activo, controladores de dominio, software de terceros como CRM, productos de uso interno para las labores de soporte a producción, etc. Para poder disponer de una granja de servidores virtuales Google provee Compute Engine [pGCP09], mientras que Amazon ofrece un producto equivalente: Elastic Compute Cloud [pAWS10].

Balanceadores de carga

Alta disponibilidad, escalabilidad, optimización de recursos y seguridad son conceptos sin los cuales no podría ofrecerse un servicio de calidad. Para conseguir estos objetivos se utilizan los balanceadores de carga. Éstos se colocan delante de un conjunto de terminado de servidores, que comparte una misma funcionalidad, y reparten la carga de trabajo entrante entre ellos. Google dispone de Load Balancing [pGCP10] como mecanismo de balanceo de carga, y Amazon de Elastic Load Balancing [pAWS11].

DNS

En un ecosistema dinámico donde coexistan múltiples máquinas prestando servicio es necesario utilizar el protocolo DNS. Para ello existen Google Cloud DNS [pGCP11] y Amazon Route 53 [pAWS12].

Integración VPN

Existen situaciones en las que es deseable que un conjunto de servidores virtuales contratados a un proveedor se integren en la LAN de una organización. La solución ofrecida por los proveedores cloud es la de la creación de redes VPN entre un segmento de su infraestructura y la red propiedad del cliente. Aquí entran en juego Google Interconnect [pGCP12] y Amazon Virtual Private Cloud [pAWS13].

4.5.3 Parte privada

Diseñar una nube privada es una labor conceptualmente sencilla, pero no así su construcción real. Para crear una nube privada se requieren varios elementos fundamentales.

Equipamiento

De forma similar a un Centro de Procesamiento de Datos convencional (CPD), son necesarios dos tipos de equipamiento. Por un lado, una granja de lo que comúnmente se denominan ‘maquinas desnudas’. Una máquina desnuda, o bare-metal, es un servidor sin sistema operativo. Las especificaciones de los mismos dependerán del uso final que se les

vaya a dar (almacenamiento, cómputo, servidor de aplicaciones, DNS, etc.). En ellos se ejecutará un software que se denomina monitor de máquina virtual o hypervisor [WIKI26] cuya funcionalidad es la de ejecutar máquinas virtuales. Por otro lado, y no menos importante, también es necesario disponer de equipamiento de red que interconecte la granja de servidores.

Imágenes de máquinas virtuales

Una máquina virtual se compone de dos partes. La primera es una imagen de un sistema operativo funcional configurado de una determinada manera, que se puede ejecutar en un hypervisor. La segunda es una definición del hardware virtual de que dispone dicha máquina. Existen tantos tipos de máquinas virtuales como configuraciones de máquina física se puedan idear. De entre las más comunes podemos mencionar los servidores de aplicaciones, de bases de datos o de almacenamiento.

Software de orquestación

Una granja de servidores virtuales, pese a tener múltiples ventajas, carece de auto-aprovisionamiento bajo demanda, no es escalable ni elástica, ni es cuantificable por completo. Por tanto, no compone por sí sola una nube. Para conseguir una nube, es necesario disponer de un software que gestione los hypervisores y máquinas virtuales y que monitorice la carga y despliegue, reubique o retire instancias virtuales, para cumplir con una determinada calidad de servicio. La naturaleza de este tipo de software tiene una complejidad muy alta y, por ello, no existen muchos proveedores con productos adecuados para el entorno empresarial que nos atañe. Podemos mencionar los cuatro más importantes: Citrix CloudPlatform [pCTX01], VMware vCloud Suite [pVMW01], Microsoft System Center [pMSF01] y Apache CloudStack [pACS01].

Otras consideraciones

Para componer una nube híbrida hay que enlazar la parte pública y privada. La práctica más común es la de utilizar una VPN para integrar la parte pública en la red de la privada. Sobra decir que para el correcto diseño y mantenimiento de una nube privada, es necesario disponer de personal altamente cualificado.

Adicionalmente, la nube privada se puede integrar con servidores físicos convencionales. Existen ciertas recomendaciones al respecto [TECHR1]. Básicamente, esto se hará debido a limitaciones en las licencias, en aquellos servidores críticos cuya virtualización no se haya probado con la suficiente fiabilidad o aquellos que dependan directamente del hardware físico subyacente.

4.6 Arquitectura de la red de oficinas

Anteriormente se expusieron los modelos actuales de despliegue del EdT y cuáles eran las principales carencias de cada uno. Una vez presentada la arquitectura global de la entidad cabe presentar la que regirá el funcionamiento de las oficinas.

En la Ilustración 9, podemos ver el esquema general de dicha arquitectura. Por un lado tenemos los puestos de trabajo de los empleados. Éstos, como ya se especificó en el planteamiento de la solución, serán equipos 2 en 1. El objetivo es facilitar la movilidad del empleado entre las distintas áreas de la oficina, ofrecer una imagen de actualidad tecnológica y perfilar una forma de interacción con el cliente. El uso de este tipo de ordenadores impone el establecimiento de una red wifi en la oficina que habrá que proteger apropiadamente. La principal herramienta de trabajo del empleado, el EdT, será convenientemente actualizado. Se migrará del concepto actual de aplicación pesada a una aplicación web.

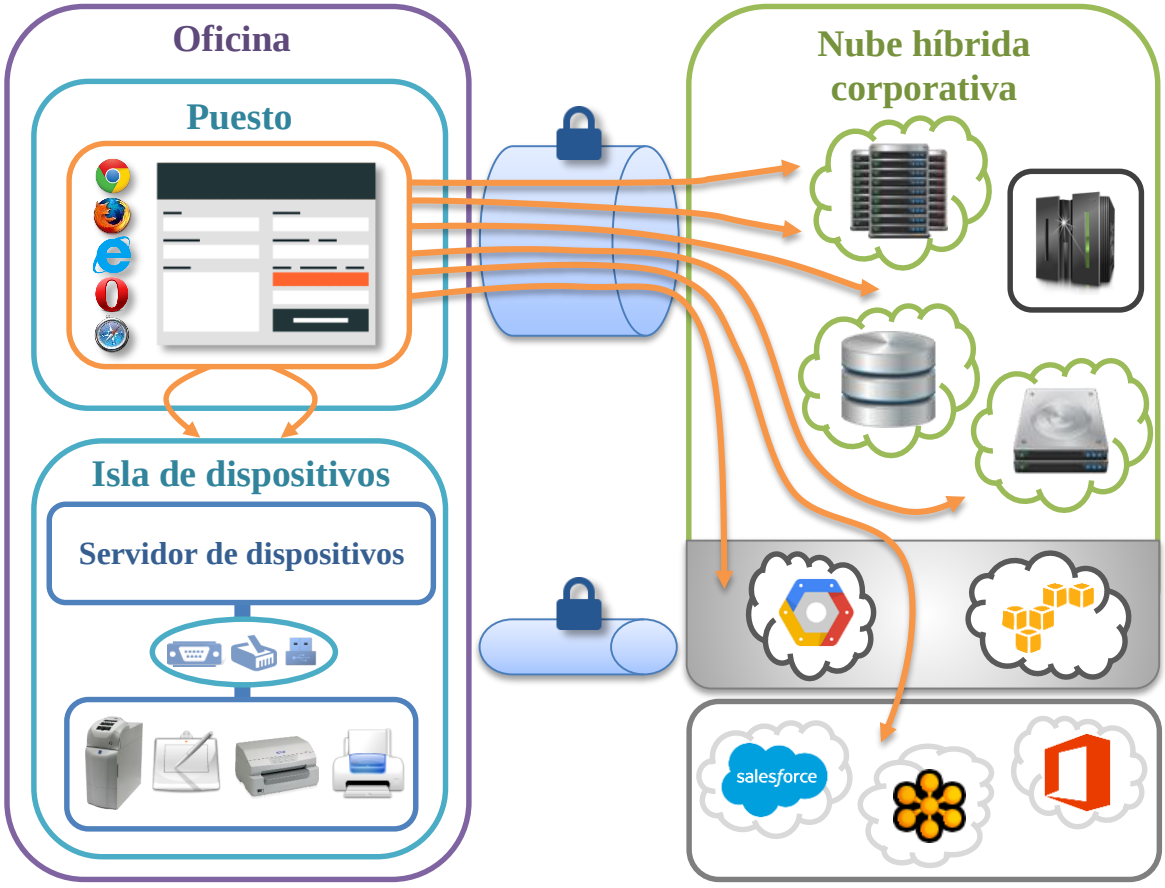

Ilustración 9 - Arquitectura del nuevo Escritorio de Trabajo

El segundo componente clave de las oficinas es la Isla de Dispositivos. Éstas proveen acceso remoto a la periferia bancaria de la oficina. Para ello constan de un servidor que encapsula toda la lógica necesaria para atender las peticiones de servicio y comunicarse con los clientes, la NHC y servicios de terceros.

En lo que resta de capítulo, abordaremos con más detalle estas dos piezas: su diseño, las implicaciones de su uso y los beneficios que se desprenden de estas elecciones.

4.7 Escritorio de Trabajo

En efecto, el nuevo EdT será una aplicación web. Más concretamente, una RIA (Rich Internet Application). Una RIA es una aplicación que se descarga y ejecuta mediante un navegador web [WIKI27]. Existen varias posibilidades como pueden ser Flash, Silverlight JavaFX, HTML/Javascript, etc., cada una con sus pros y contras. Independientemente de qué familia de tecnologías se utilice, el uso de RIAs aporta ciertos beneficios.

Despliegue

Uno de los puntos a favor del uso de una RIA radica en que esta arquitectura convierte, indirectamente, a los servidores de aplicaciones en nodos de distribución. Esto se debe a que en la primera conexión, el navegador se descarga el código de la aplicación. Ésta, a su vez, puede descargar código adicional en base a las necesidades puntuales. Dado que los navegadores tienen la capacidad de cachear muchos de los elementos descargados en sesiones anteriores, se simplifican enormemente las tareas de distribución. Asimismo, es habitual que los inicios de sesión se distribuyan en un intervalo horario y, también, que no todos los usuarios accedan a las mismas funciones a la vez. Esta aleatoriedad en el acceso y carga de funciones, sumado a la capacidad de cacheo, reduce considerablemente la carga a la que se somete a los servidores en los momentos en que los clientes se descarguen la RIA, o partes de ella.

Distribución de la carga de trabajo

Una RIA normalmente reparte la carga de trabajo entre el cliente (navegador) y los servidores. Esta división de trabajo depende en gran medida de la arquitectura software utilizada. Pero normalmente, la interfaz de usuario y toda su actividad de procesado se colocan en el lado cliente, mientras que en el servidor se ubican todas las tareas relativas a la operación y manipulación de datos.

Consumo de ancho de banda durante la ejecución

El reparto de funciones arriba comentado, permite al cliente encargarse de las actividades locales, pequeños cálculos, maquetación de la interfaz, gestión de eventos y animaciones y demás. De esta manera, se reduce la cantidad, volumen y frecuencia en las comunicaciones entre cliente y servidor en comparación con las arquitecturas de cliente ligero-servidor pesado.

Interfaces de usuario más ricas

Por norma general, las interfaces de usuario son potencialmente más ricas que sus versiones pesadas de escritorio, a la par que son más responsivas que las aplicaciones web convencionales. Además, permiten hacer diseños de una sola página lo que elimina las transiciones de carga y aumenta la transaccionalidad de las aplicaciones. Por último, algunas de las plataformas permiten su uso en terminales móviles [EZIN1].

Desventajas

La práctica totalidad de plataformas RIA implica la instalación de una pieza de software que actúe como entorno de ejecución. Esto implica que se debe distribuir y mantener un software adicional en los puestos. En cuanto a la seguridad, estos entornos suelen usar técnicas de sandboxing para aislar la aplicación del sistema operativo. Sin embargo, con el tiempo se ha demostrado que la existencia de estas piezas supone graves riesgos de seguridad. Adicionalmente, en algunos casos, la pobre implementación de estas plataformas deriva en un consumo de recursos exagerado.

Conclusión

Las ventajas del uso de una RIA son evidentes. Sin embargo, existen múltiples plataformas para ello. Pero de entre todas ellas se utilizará el Javascript, en conjunción con HTML y CSS. Esta elección se justifica por varios motivos:

- Tanto Flash como Silverlight (las de mayor aceptación) son tecnologías que van perdiendo tasa de adopción poco a poco; justo al contrario que le ocurre al trio HTML5, CSS3 y Javascript[4].
- Bajo los criterios de onmicanalidad, usabilidad y homogeneidad en las interfaces de usuario, el EdT debe ser similar al portal de acceso (o aplicación móvil) de los clientes. El uso de Flash o Silverlight o JavaFx implica la instalación de un software en el equipo en que vaya a ejecutarse. Y no se tiene la certeza de que el usuario disponga de él o quiera instalárselo.
- Para el uso de Javascript no es necesario distribuir ninguna pieza software adicional pues su entorno de ejecución es el navegador web en sí mismo. Por lo tanto, basta con que se disponga de un navegador moderno. Lo cual ya es casi una obviedad en sí misma.
- A día de hoy Javascript se ha convertido en un estándar de facto en lo que a ejecución el lado cliente se refiere. Y su índice de penetración es elevadísimo. Prueba de ello es que se utiliza en prácticamente la totalidad de aplicaciones web, incluidas las de los

[4] A fecha 31 de Agosto de 2015. Fuente: https://www.google.com/trends/explore#q=adobe+flex,silverlight,java+applet,HTML5,javascript.

mayores proveedores de servicio. Por lo tanto, su elección garantiza una mayor longevidad de la solución; así como acceso a personal cualificado, contribuyendo a mantener unos costes acotados.

La elección de una RIA Javascript, soluciona los problemas de los anteriores esquemas de EdT. Por un lado, se reduce el consumo de ancho de banda entre la oficina y la NHC. Además, se simplifican y agilizan las tareas de distribución. Lo que, a su vez, elimina la necesidad de servidores de distribución o virtualización. Y, al ser tecnologías libres con numerosos elementos y servidores de código abierto, se minimiza el coste de licencias.

4.7.1 Lenguajes y bibliotecas de programación

En la actualidad existen tantos lenguajes de programación que difícilmente se podrían exponer y diferenciar los beneficios y desventajas de usar uno u otro. Por un lado, es necesario el uso de varios tipos de lenguajes: Uno para la visualización estática de la información, otro para dotar a dicha capa de dinamismo y finalmente uno adicional que aporte la lógica de la aplicación y se ejecute en la parte servidora.

4.7.1.1 Visualización

En este apartado se recurrirá a HTML (HyperText Markup Language) para la parte estructural y CSS (Cascading Style Sheets) para el aspecto y formato. Ambos lenguajes existen de forma simbiótica y son a día de hoy la base estandarizada de programación web. Este último punto es importante debido a que al ser lenguajes muy extendidos, es fácil, rápido y barato tener acceso a personal con la formación necesaria.

Respecto a qué versión utilizar, dada la naturaleza de la aplicación que se va a desarrollar, es necesario el uso de HTML5 y CSS3. Si bien es cierto que las anteriores iteraciones de ambos lenguajes aportaban características nuevas, el salto cualitativo en estas últimas versiones es enorme.

Entre las características de HTML5 y CSS3 [MOZ1][MOZ2][W3O3], se ha añadido soporte nativo para reproducción de archivos multimedia, gráficos 2D y 3D, almacenamiento en el lado del cliente, drag&drop, geolocalización, manipulación de ficheros, mensajería entre documentos de distintos orígenes, base de datos local al cliente, formulaciones matemática con MathML, etc. Como se puede apreciar, una aplicación con las características del EdT requiere el uso de esta versión de HTML. En lo referente a CSS3, las novedades no son tan importantes, pero se integra con HTML5 y permite crear interfaces de usuario ricas y vistosas. Algunas de las nuevas funcionalidades son: mayor soporte a transformaciones y animaciones, mejoras en la gestión de los fondos, efectos de texto y maquetaciones multicolumna.

Pese a ser versiones relativamente recientes su soporte en los navegadores web actuales es muy amplio[5] y su uso garantiza una mayor longevidad a la aplicación.

4.7.1.2 Dinamismo en el cliente

A día de hoy JavaScript se integra con HTML y CSS de tal forma que difícilmente podría entenderse la web si la conjunción de los tres. En torno a JavaScript ha proliferado la creación de multitud de bibliotecas (frameworks) que aportan nuevas funcionalidades y simplifican la creación de aplicaciones complejas. Sirvan como ejemplos jQuery [pJS01], Dojo [pJS02], AngularJS [pANGJS], Kendo UI [pJS03], Ember.js [pJS04], React.js [pJS05], MooTools [pJS06], Socket.IO [pJS07], y un largo etcétera.

Sin embargo, los requisitos del proyecto exigen el uso de un framework potente diseñado específicamente para la creación de RIAs. Aquí las opciones son menores destacando: AngularJS, Backbone.js [pJS08], Ember.js o React.js. Cada una de ellas tiene sus pros y sus contras pero buscamos una que tenga buena proyección de futuro, muy extendida y con un amplio soporte en la comunidad de desarrolladores. AngularJS cumple a la perfección con todos estos requisitos [JSRIA1][JSRIA2].

AngularJS es un framework de código libre para aplicaciones web que es mantenido actualmente por Google. Su objetivo es el de la creación de aplicaciones de página única. Entre los muchos beneficios que aporta, podríamos destacar:

- Implementa el modelo Model-View-Controller (MVC) [WIKI28] directamente en el cliente. Ello permite segmentar de forma clara el código dedicado a visualización (interfaz de usuario), el modelo (datos de la aplicación) y de control (lógica de la aplicación). También soporta el modelo Model-View-Viewmodel (MVVM) [WIKI29] que es una evolución del MVC que simplifica el controlador acoplando el modelo directamente con la vista.
- Incorpora un entorno de ejecución de test. Dado que javascript es una tecnología difícil de testear, esta característica supone un avance en cuanto a estabilidad y fiabilidad de la aplicación resultante.
- Enlaces bidireccionales entre componentes DOM y el modelo, lo que implica reducir drásticamente la cantidad de código dedicado a ese trasvase de datos.
- Incorpora soporte para plantillas HTML que se completan después con los datos del modelo. Esto contribuye a mejorar el flujo de trabajo entre diseñadores y desarrolladores.

[5] A fecha 31 de Agosto de 2015. Fuente: https://html5test.com/results/desktop.html.

- Implementa un sistema de inyección de dependencias que ahorra mucho código declarativo y posibles fallos de dependencias.
- Se contempla la extensión del catálogo de etiquetas disponibles mediante un mecanismo que se denomina directivas. Esto permite crear elementos complejos o extender los actuales incluyendo un comportamiento por defecto. Estas directivas pueden usarse en el documento como si fuesen etiquetas convencionales. Esta característica aumenta drásticamente la reusabilidad de código.

4.7.1.3 Lógica del servidor

En la actualidad existen multitud de opciones que permitan trabajar con servicios REST como pueden ser PHP, Java, Ruby, Dart, Python, .NET, etc. Sin embargo, dado que vamos a usar los servicios en la nube de Amazon y Google deberemos ajustar nuestra decisión a aquellos lenguajes y plataformas soportados por ellos. A día de hoy Google Cloud Platform dispone de librerías cliente escritas en Java, Javascript, .Net, Objetive-C, PHP y Python de forma estable. Mientras que Amazon tiene APIs en Java, Javascript, .Net, PHP, Ruby y Python.

.NET y Objetive-C quedarían descartados debido a las limitaciones de sistema operativo que imponen (Windows y OS X respectivamente). De entre las opciones restantes elegiremos Java por varias razones. En primer lugar, su madurez y cobertura en escenarios SOA y REST es mayor que en el resto. Asimismo, Java dispone de un ecosistema de librerías (son especialmente interesantes las que dan acceso a código nativo y periféricos), servidores y herramientas mucho más rico. Finalmente, y continuando con el razonamiento de los puntos anteriores, elegir una herramienta estandarizada y ampliamente extendida facilita el acceso a personal cualificado y garantiza la estabilidad y el ahorro de costes a largo plazo.

4.8 Islas de Dispositivos

Una de las mayores limitaciones del EdT consiste en que, al ser una aplicación que se ejecuta en el navegador, no puede ejecutar código del sistema operativo. Esta restricción impide que pueda acceder a dispositivos. Como vimos en el capítulo anterior, se incluyen las islas de escaneo para superar dicha condición. Adicionalmente, la concepción de un nodo de dispositivos accesible desde cualquier punto de la oficina aporta numerosos beneficios adicionales. Como, por ejemplo, el aumento de la disponibilidad del servicio y, por tanto de la productividad, así como un importante ahorro de costes tanto directos (adquisición) como indirectos (mantenimiento). En esta sección desgranaremos todos los detalles al diseño de las IdDs.

4.8.1 Características

Una IdD debe contener todos los elementos necesarios para que pueda prestar el servicio exigido. Para ello será necesario que disponga de las bibliotecas y drivers que permitan el acceso a los distintos periféricos, un medio físico que permita a los usuarios identificar una IdD concreta y, obviamente, un catálogo de servicios REST, a ejecutar dentro de un contenedor apropiado, que será consumido desde el EdT.

4.8.2 Identificación

En un escenario habitual, existirá más de una IdD en la oficina y los periféricos de todas ellas estarán disponibles para el gestor a través del EdT. Sin embargo, es necesario proveer un sistema de identificación para que el gestor sepa en cada momento el dispositivo de qué IdD concreta va a utilizar y pueda seleccionar otro en caso necesario.

La forma más sencilla de hacer esto es la de dar un nombre de máquina determinado a cada IdD (por ejemplo: islaA, islaB, etc.), referir la nomenclatura de los dispositivos a dicho nombre (impresora@islaA) y colocar un cartel con el mismo en la IdD para que el gestor pueda identificarlas entre sí. Sin embargo, es una solución que no termina de encajar con el escenario de modernidad que se plantea y que no está exenta de ciertos problemas a largo plazo, como pueden ser la substracción, pérdida o deterioro del cartel, cambio de nombre de una isla, etc.

Ilustración 10 - Dispositivo led RGB por USB

Existe una forma más apropiada de acometer esta tarea: incluir un dispositivo en la IdD que le permita publicar su identidad de forma autónoma. La opción más obvia sería una pequeña pantalla LCD. Pero esto implica que el usuario se traslade hasta la IdD para conocer su identidad. Sin duda un inconveniente menor pero que se resolvería mediante el

uso de un módulo USB con un led RGB como el que aparece en la Ilustración 10. El uso de este tipo de dispositivos permite fijar distintos colores, por ejemplo en base a la IP, que son fácilmente reconocibles a distancia. Adicionalmente, se puede utilizar para señalizar situaciones de error mediante el uso de determinados colores, patrones de intermitencia o una mezcla de ambos. Existen varios modelos pero el elegido es el blink(1) [pBLINK] principalmente porque dispone de un API Java para su manejo.

Además, el gestor deberá poder seleccionar un periférico de cada familia para su uso por defecto o, incluso, agrupar cierto conjuntos de ellos bajo un nombre con el fin de poder cambiar de uno a otro en función de la ubicación donde se encuentre (área de trabajo, de cliente o privada).

4.8.3 Servicios

Catálogo

El catálogo de servicios que ofrezca una IdD dependerá en gran medida de los dispositivos que en ella se vayan a ubicar. Sin embargo, sean cuales sean, su API debe ser lo más atómica posible para minimizar el tráfico y mejorar la transaccionalidad de las operaciones.

Contenedor

Para servir las páginas y código necesarios para ejecutar la aplicación web es necesario un servidor. Existen multitud de ellos pero, dado que la lógica del servidor se programará en Java, la lista queda algo acotada. Para reducir la lista y los costes del proyecto utilizaremos uno de código libre. Llegados a este punto tenemos varias opciones disponibles: Jetty [pJCon1], Apache Tomcat [pJCon2], TomEE [pJCon3], Glassfish [pJCon4], Resin [pJCon5], WildFly [pJCon6], etc. Elegiremos Tomcat de entre todos ellos por ser uno de los más utilizados [TOM1], por su fiabilidad y escalabilidad y porque al ser un contenedor de servlets, y no un servidor de aplicaciones al uso, tiene un menor consumo de recursos.

4.8.4 Acceso a dispositivos

A día de hoy, la práctica habitual de los fabricantes de dispositivos es la de escribir los drivers en código nativo. Por otro lado, Java dispone de innumerables librerías que extienden su funcionalidad. Muchas de ellas están orientadas a permitir el acceso y control de distintas familias de periféricos. Las familias de dispositivos que se utilizarán en las IdDs son principalmente dispositivos financieros, de adquisición de imágenes y de impresión. Éstos últimos son soportados directamente en Java. Pero veamos las soluciones que se adoptarán para los otros casos.

4.8.4.1 XFS (eXtensions for Financial Services)

El acceso a dispositivos de periferia bancaria es un apartado limitado y controvertido. Al ser periféricos muy concretos dedicados a cubrir un nicho de mercado con un tipo de cliente muy específico, lo común hasta los años 90 era depender de las librerías de acceso de cada fabricante. Lo que obligaba a hacer desarrollos específicos para cada uno de ellos, e incluso para cada modelo concreto de dispositivo.

En los años 90 Microsoft, inicia un proceso de estandarización denominado WOSA/XFS [WIKI30]. Años después, el CEN (Comité Européen de Normalisation) crea su propio estándar basado en el anterior denominado CEN/XFS [WIKI31]. Sin embargo, ambos están limitados a su uso en plataformas Windows. Para superar dicha limitación, a finales de los años 90, se funda el J/XJS Forum con el claro objetivo de crear una solución XFS multiplataforma. Entre los fundadores se encuentran fabricantes (DeLaRue, NCR y Wincor) e integradores (IBM y Sun Microsystems).

Paralelamente, se desarrolla JavaPOS (Java for Point of Sale Devices) [WIKI32]. JavaPOS es una API Java enfocado al manejo de periféricos de puntos de venta como pinpads, lectores de banda magnética, códigos de barras, etc. El esfuerzo del JXF Forum se centra en crear una API Java que aúne las funcionalidades de XFS y JavaPOS. En el año 2000 se libera la primera versión de dicha API que se denomina J/XFS (Java eXtentions for Financial Services) [WIKI33]. Actualmente, J/XFS es mantenido por el CEN pero desde 2002 sólo se han liberado dos nuevas versiones.

J/XFS se basa en un modelo cliente-servidor en el que cada dispositivo está representado por un DS (Device Service). Los DS son específicos de cada fabricante e implementa, mediante JNI, una serie de funciones concretas muy detalladas declaradas para su familia de dispositivos correspondiente. Aquí reside el principal problema de J/XFS: la API se definió de forma que pudiese servir para acceder a multitud de periféricos de naturalezas dispares. Como resultado, se tiene una API demasiado genérica y extensa que la hace difícil de interpretar e implementar correctamente. A la par, los fabricantes siguen innovando y dotando a sus productos de nuevas funcionalidades que no están contempladas en el estándar, ya obsoleto. Para complicar aún más las cosas, es frecuente que los clientes pidan funciones o comportamientos específicos a los proveedores de los DS. A menudo, la salida a esta encrucijada reside en una serie de métodos contemplados para los DS que son, en realidad, puertas de atrás. Mediante estos mecanismos se implementa cualquier tipo de funcionalidad que el fabricante o el cliente desee. Este tipo de prácticas ha derivado en que J/XFS no sea un estándar como tal si no una base sobre la que, cada cliente o integrador, desarrolla su solución de acceso a periféricos.

En los últimos años, diversos integradores están promoviendo iniciativas como Xpeak [WIKI34] que pretenden emular el comportamiento y beneficios de J/XFS a unas tecnologías más actuales. Sin embargo, en muchos casos no son más que una capa de servicio web sobre un J/XFS convencional. Por lo que heredan los numerosos inconvenientes del mismo.

4.8.4.2 Tecnologías de adquisición de imagen

Es evidente la necesidad de poder digitalizar documentos físicos. Para ello se usarán dispositivos de captura de imagen (normalmente escáneres) o equipos multifunción de gran formato. De estas dos opciones, la que nos importa en este punto es la primera (pues la segunda suele ser un ecosistema cerrado).

Típicamente existen cuatro APIs para poder acceder y manejar un escáner. Analizaremos brevemente cada uno de ellos:

- TWAIN [pDEV1]: es el más antiguo de todos y por ello el más anticuado tecnológicamente hablando pero también el más extendido. Se ha publicado bajo licencia LGPL por lo que su uso es libre y no es necesario publicar el código fuente de las aplicaciones que lo utilicen. Desde el año 2008 soporta plataformas Windows, OSX y Linux.
- SANE [pDEV2]: Una versión similar a TWAIN pero más avanzada tecnológicamente. Está publicado con licencia GPL lo que obliga a publicar el código fuente de toda aplicación que haga uso de él. Pese a que soporta plataformas Windows, OSX y Linux, sólo está especialmente extendido en entornos Linux.
- ISIS [pDEV3]: es un estándar libre desarrollado por EMC Captiva. Sin embargo sólo existen librerías de desarrollo para C++ y .NET. Lo que limita su uso.
- WIA [pDEV4]: es un API propiedad de Microsoft que sólo se soporta en entornos Windows.

Una vez analizados los pros y contras de cada opción nos decantaremos por el uso de TWAIN por estar muy extendido y soportado en todos los sistemas operativos de interés.

No existen muchas bibliotecas Java para soportar TWAIN. Básicamente se reducen a dos, ambas de pago (aunque con versión de prueba gratuita): JTwain de Asprise [pTWA1] y Morena de Gnome [pTWA2]. Elegiremos la más extendida, Morena, dado que será más sencillo acceder a soporte y recursos de interés.

4.8.4.3 JNI (Java Native Interface)

Para todo aquel dispositivo que no entre en los grupos anteriores, dependeremos de que el fabricante disponga de un SDK en Java o utilizar directamente las funciones nativas mediante JNI [pJNDI].

En los primeros años de existencia de Java, cuando las tecnologías web todavía estaban arrancando, los responsables del desarrollo de Java se dieron cuenta de que necesitaban que Java fuese capaz de entenderse con código nativo escrito principalmente en C y C++.

Esta posibilidad abría la puerta para que Java fuese capaz de interactuar con el sistema operativo y con librerías y programas ya existentes y de amplia difusión. Así mismo, aunque en menor medida, permitía optimizar el uso de Java en aplicaciones o rutinas en las que otro tipo de lenguaje fuese más apropiado; como por ejemplo C y el procesamiento matemático.

JNI provee un puente bidireccional entre el código Java y el nativo implementando una serie de funciones en código nativo y unos mecanismos de mapeo de tipos de datos. Sin embargo, existen ciertos inconvenientes en el uso de JNI [WIKI35]:

- Errores en el uso de JNI pueden desestabilizar la JVM (Java Virtual Machine). Esos fallos son muy difíciles de reproducir y solventar.
- Una aplicación Java que use JNI pierde su componente de portabilidad dado que depende de funciones concretas del sistema que no se implementan en la JVM.
- La JVM no dispone de los mecanismos necesarios para liberar los recursos utilizados en el código nativo. Por lo tanto, es necesario asegurarse de que dicha labor es llevada a cabo en la parte nativa del programa.

Existen algunas limitaciones adicionales más concretas en las que no se va a entrar en detalle. Si el lector está interesado puede consultar las referencias

4.9 Seguridad en los accesos

El acceso a recursos y servicios de la entidad desde el exterior de la red se protegerá de manera convencional mediante el uso de firewalls, mecanismos de autorización y/o autenticación, antivirus, correcta asignación de servidores a zonas DMZ, etc. Sin embargo, las oficinas son un caso especial por ser elementos de la entidad que están deslocalizados. Para proteger las oficinas se propone un esquema de tres capas que se detalla a continuación.

Como punto de partida, hay que evitar que un tercero, fuera de las instalaciones de la entidad, pueda escuchar las comunicaciones. Para ello se utilizará una VPN que enlace el router de la oficina con la NHC. VPN establece un canal seguro punto a punto a través de una WAN (Wide Area Network). Una de las mayores limitaciones de VPN consiste en que no es cien por cien eficaz conectando dominios de broadcast. Esto deriva en que los protocolos basados en mensajes broadcast pueden no funcionar correctamente. Si esto fuese una limitación habría que recurrir a proteger el canal mediante una VPLS; que es una variante de VPN que emula la funcionalidad completa de una LAN sobre un canal seguro.

Sin embargo, el mayor riesgo de este escenario radica en la existencia de una red wifi en las oficinas. Pues en caso de que un atacante consiguiese vulnerar la seguridad de la red inalámbrica, tendría acceso directo al CPD. Para solventarlo, en primer lugar se forzará a que, en la medida de lo posible, todas las comunicaciones vayan cifradas por TSL o similar.

En segundo lugar se implementará, para el EdT y las IdDs, sistemas de autenticación combinada.

Asegurando el EdT

Los esquemas de usuario-contraseña son suficientes en cierto tipo de escenarios. En el que nos atañe, sin embargo, es necesario añadir una capa extra de seguridad. Para que el nivel de seguridad sea óptimo se tendrá que autenticar tanto el usuario como el dispositivo.

Para la autenticación del empleado se utilizará usuario-contraseña. Para la del navegador, se empleará una técnica denominada Browser Fingerprinting [PAN1] [CHR1] para generar un código que lo identifique de forma única (***BF***). El registro de un navegador en el sistema se validará mediante ***OTP***. Tras este proceso genera un token de renovación (***TR***) que, a su vez, permite obtener un token de sesión (***TS***). Todas las peticiones subsiguientes deberán incluir tanto el ***TS*** como el ***BF***. Una entidad de autenticación (***SA***) registrará los tokens y validará los accesos. La Ilustración 11 muestra todo el proceso de autenticación desde el principio.

El proceso completo de autenticación se compone de tres bloques:

- Bloque I: Registro y asociación de un ordenador. Durante el primer inicio de sesión del ***USER1***, se comprueban sus credenciales y se detecta que éste no dispone de un ***TS***. Se prepara una transacción de autenticación ***T1*** para ***USER1***. A continuación, se redirige el navegador contra Servidor de Autenticación, ***SA***, para procesar ***T1***. El ***SA*** comprueba que la petición es correcta y envía un código ***OTP*** al teléfono del usuario. Una vez introducido y comprobado el código, ***SA*** crea un ***TR*** y almacena la tupla ***{USER1, BF, TR}***. El navegador, a su vez, almacena ***TR***. A partir de éste momento, se ha registrado el dispositivo para su uso por parte de ***USER1***. Esta negociación sólo es necesaria hacerla la primera vez que un usuario inicia sesión desde un equipo nuevo. A partir de ese momento, todo se gestionará de manera automática en base al ***BF*** y al ***TR***.
- Bloque II: petición de un token de sesión. Durante el arranque, el ***EdT*** es consciente de que no dispone de un ***TS*** pero sí de un ***TR***. Así que contacta con ***SA*** y le solicita un ***TS*** presentando como credenciales el ***TR*** y el ***BF***. ***SA*** las comprueba y genera un ***TS*** que devuelve a la aplicación. El tiempo de expiración de los ***TS*** es de

un día, por lo que esta negociación se llevara a cabo diariamente durante el primer inicio de sesión del usuario.

- Bloque III: inicio de sesión normal. El ***EdT*** dispone de un ***TS***. Así que compone una petición con el usuario-contraseña de ***USER1***, el ***BF*** del navegador y el ***TS***. El servidor de ***EdT*** autentica al usuario y solicita al ***SA*** la verificación de la tupla ***{USER1, BF, TR}***. En caso afirmativo se inicia la sesión del usuario.

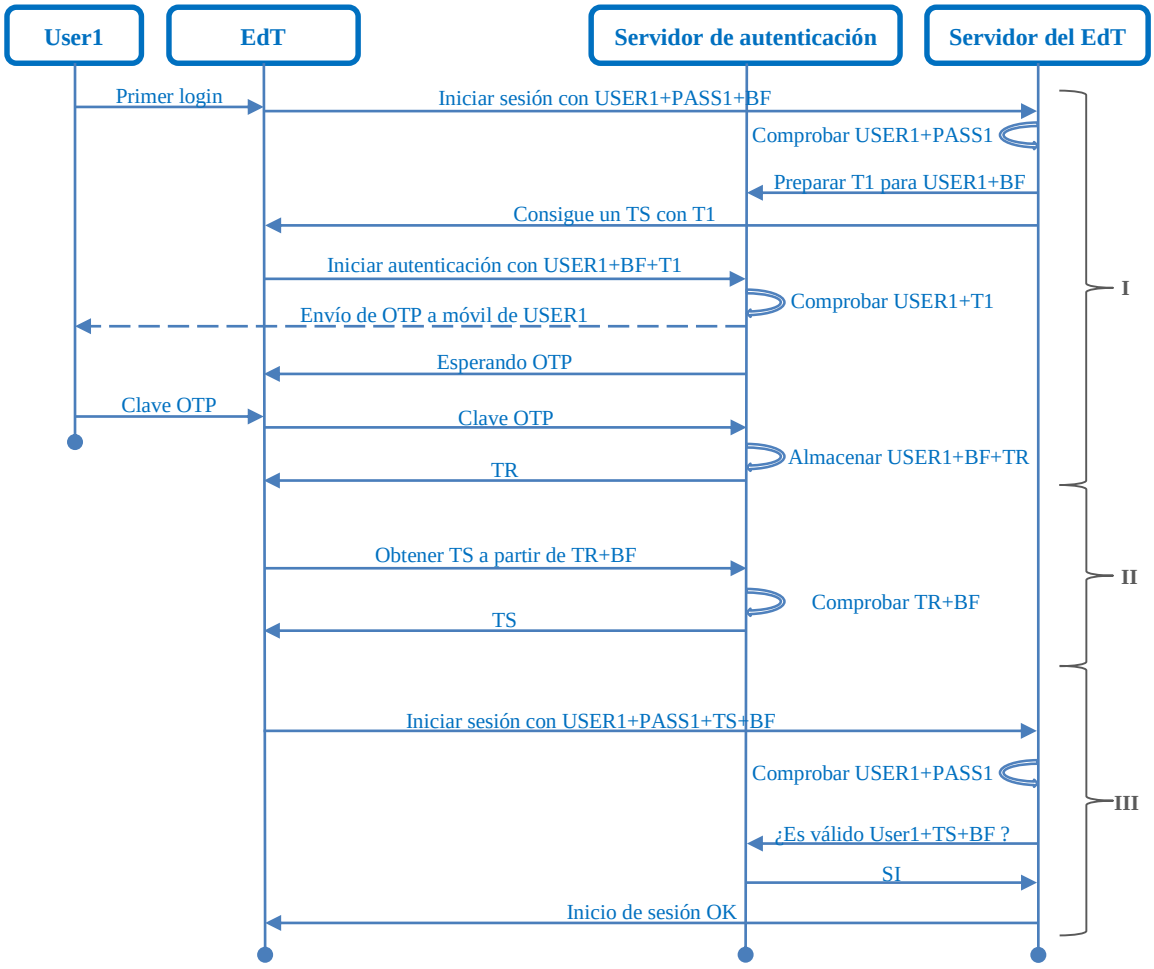

Ilustración 11 - Seguridad en el acceso para el Escritorio de Trabajo

Es posible que, a lo largo del tiempo, el BF cambie. Si eso ocurriese tendría que llevarse a cabo todo el proceso desde el bloque I. Además, todo este proceso es susceptible de ataques de tipo "man-in-the-middle" [WIKI36]. Pero como hemos dicho antes las comunicaciones están triplemente cifradas (TSL+Wi-Fi+VPN) por lo que es un riesgo menor.

Acceso desde las IdDs

El caso de las IdDs es similar al anterior pero la pieza a autenticar no es un navegador y tampoco se dispone de un terminal móvil en el que poder recibir OTPs. En su lugar,

durante la instalación, se utilizará un token criptográfico RSA que hará las veces de OTP. En sustitución del BF se utilizará un Hardware ID del dispositivo, que bien puede ser el número de serie de la placa base o similar.

Acceso a los servicios de las IdDs

Para proteger el acceso a las IdDs desde los EdTs es necesario que las primeras comprueben las credenciales de los segundos. Esto se hará mediante una simple consulta al servidor de autenticación como se muestra en la Ilustración 12.

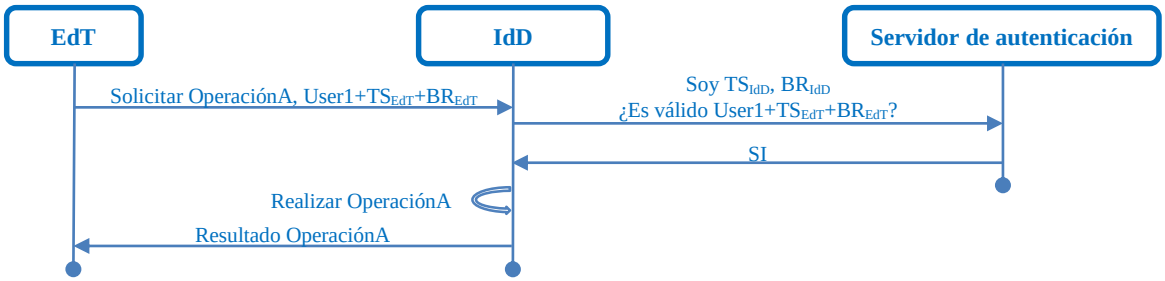

Ilustración 12 - Seguridad en el acceso a las IdDs desde los EdTs.

4.10 Alianzas y productos de terceros

El número de dependencias que tiene una entidad bancaria para poder desarrollar su actividad es elevado. Sin embargo, es importante que no se pierda el enfoque de los procesos de negocio. Para ello es recomendable contratar aquellos productos y servicios que sean necesarios pero no formen parte del núcleo operativo de la entidad. Asimismo, puede ser recomendable establecer alianzas con terceros especializados en áreas de interés en las que la empresa no quiera involucrarse directamente o no disponga de personal cualificado.

Por poner unos ejemplos, se pueden requerir productos convencionales similares a soluciones ofimáticas, correo o almacenamiento en la nube, o productos más específicos como GoToMeeting, SalesForce, BaseCamp, etc. Sea cual sea el caso, hay que limitar las relaciones con terceros a aquellos que tengan la certificación Safe Harbour.

Por último, hay que tener en consideración que este tipo de contratos y alianzas supone un incremento de tráfico entre la NHC y la infraestructura del tercero. Para que la integración y la calidad de servicio sean las adecuadas habrá que analizar cada caso de forma independiente. Pues puede ser necesario disponer líneas de comunicaciones específicas, integrar redes por VPN o ubicar ciertos procesos en la parte pública o privada de la NHC.

Capítulo 5

Prueba de concepto

5.1 Introducción

En los capítulos precedentes se ha expuesto de forma general la situación actual del sector de banca minorista, y más particularmente el de su red de oficinas. Se ha planteado un cambio conceptual en la forma en que éstas operan y dan servicio. Asimismo, se ha propuesto una arquitectura que soporte los procesos presentes y futuros y que establezca las bases para plantar cara a los nuevos competidores del sector. Llegados a este punto, se ha decidido incluir un capítulo adicional que, de forma meramente ilustrativa, muestre cómo funcionaría una operativa habitual bajo el modelo propuesto.

5.2 Caso de uso

Con el fin de demostrar cómo encajarían todas las piezas y conceptos anteriormente expuestos, se ha elegido ilustrar una operativa de contratación, en una oficina (canal presencial), de un producto por parte de una persona no cliente de la entidad.

La escena comienza con el cliente acudiendo a una cita concertada por cualquier otro canal (teléfono, web, móvil, etc.) en una oficina de su elección. El gestor asignado dispone

de dicha cita en su calendario para poder planificar el encuentro con antelación. Éste coge su equipo de trabajo, lo separa del teclado y se dirige al área de cliente para encontrarse con el cliente. Una vez juntos, el gestor explica al cliente las bondades del producto apoyado en herramientas y simuladores disponibles en su EdT. Llegados a este punto, el gestor inicia un proceso de contratación guiado a través del EdT. En primer lugar solicita el DNI al cliente y lo introduce en el escáner seleccionado por defecto. Mientras el documento se escanea, el gestor cumplimenta un formulario con los datos del cliente y se recoge la firma del mismo mediante un campo al uso en el propio EdT. Se procede al alta del cliente y del contrato. Una vez recibida la confirmación de este hecho, se puede imprimir una copia del contrato, enviarla al cliente por correo electrónico o ambas; en este caso se harán ambas. Sin embargo, instantes antes de imprimir el contrato, la impresora por defecto se estropea. El EdT detecta la situación e informa al gestor ofreciéndole una lista de impresoras disponibles en la oficina. Es este momento, el gestor elige una impresora de otra isla y el EdT redirecciona la orden de impresión.

A continuación veremos, con más detalle, los flujos y piezas involucrados en esta operativa.

5.3 Entorno necesario

Para explicar el caso de uso propuesto, se supone un entorno de oficina como el mostrado en la Ilustración 13. Concretamente, los elementos presentes en la oficina son:

- Un puesto de trabajo 2 en 1 con un navegador web moderno.
- Una IdD, con nombre de máquina ***IslaA***, en el área de cliente con un escáner y una impresora conectados.
- Una IdD, ***IslaB***, en el área de cliente con una impresora disponible.
- Una IdD, ***IslaC***, en el área de trabajo con un escáner y una impresora.
- Otra IdD, ***IslaD***, situada en el área privada con un escáner y una impresora disponibles.
- Aunque ya se detalló anteriormente en el capítulo 4, las oficinas dispondrán de una red WiFi protegida, estarán conectadas a la NHC por VPN, y contarán con un servidor DHCP propio que asigne direcciones IP dentro del rango de uso exclusivo de la oficina. Además, todas las comunicaciones utilizarán TLS.

Ilustración 13 - Entorno de oficina supuesto para el caso de uso.

En cuanto a la NHC, se requieren ciertos servicios (cloud o licenciados), bien en la parte pública bien en la privada. Son los siguientes:

- Servidor web: su uso principal será el de servir el EdT. Contendrá aquellos servicios necesarios para mantener una caché de dispositivos y atender las peticiones inherentes al proceso de contratación.
- Bases de datos: en ellos se almacenará la información relativa al perfil de cliente, los productos contratados, etc.
- Caché distribuida: su uso puede ser muy variado pero en el caso que nos atañe se utilizará para almacenar una lista de los periféricos disponibles en cada oficina.
- Almacenamiento: en el escenario actual se utilizará como repositorio documental donde almacenar los documentos identificativos del cliente y los contratos.
- Colas de mensajes: el uso de este tipo de servicios permite desacoplar, de forma asíncrona, la información generada por los publicadores de las acciones que, en base a esos datos, deban tomar

los subscriptores. Se utilizará para que las IdD informen a los servidores de los distintos dispositivos y sus estados.

- Servidores virtualizados: en ellos se alojarán procesos no convencionales que formen parte del flujo de trabajo. En nuestro caso de uso concreto, contendrán un conjunto de servidores balanceados que realicen tareas de Reconocimiento Óptico de Caracteres (ROC) [WIKI37] para extraer la fotografía y otra información del documento identificativo de cliente.

Todas estas piezas deberán estar convenientemente replicadas y balanceadas para soportar la carga de toda la red. Para que no haya merma en la calidad de servicio, la situación óptima contaría con un conjunto de reglas de monitorización y configuración que permita, de forma automática, replicar instancias de servicio una vez alcanzado un umbral determinado.

5.4 Flujos involucrados

En esta sección se mostrarán los diagramas de secuencia del proceso de estudio. Sin embargo, y con objeto de clarificar los flujos se tendrán en cuenta las siguientes premisas:

- Se considerará que el puesto del gestor ya se encuentra registrado en el sistema conforme al proceso explicado en el punto 4.8 Seguridad en los accesos.
- Asimismo, el usuario ya ha iniciado sesión previamente
- El ***TS*** no caducará durante el ejemplo, por lo que no es necesaria su renovación.
- Las partes necesarias del EdT ya se encuentran cacheadas en el navegador del puesto de trabajo.
- La caché distribuida dispone de una lista previamente cargada de pares ***{IPredOficinaA, OficinaA}***. En caso de que alguno de los pares haya caducado o no esté disponible en la caché, se recurrirá a la base de datos para obtener dicho dato y se almacenará en la caché para futuros usos.
- Para su correcta referencia, los periféricos se nombran con el patrón modelo_dispositivo@nombre_isla.
- La caché distribuida tiene, previamente cargada, una lista con los perfiles y dispositivos por defecto del gestor.

- El gestor, cuando acude a la cita con el cliente, ya tiene previamente seleccionado un perfil de dispositivos que incluye ***escaner@IslaA*** e ***impresora@IslaA***.
- Por sencillez, se obviará el envío del ***TS*** y el ***BF*** desde el EdT y la IdD a la NHC. Así como la autenticación delegada para acceder desde el EdT a los servicios de las IdDs. Ambos explicados previamente en la sección 4.8.

Por último, y para clarificar el entendimiento del proceso global, se dividirá el flujo completo en fases menores conceptualmente independientes.

Asincronía

Como se podrá apreciar en los diagramas de secuencia de los puntos posteriores, la mayoría de llamadas entre las distintas piezas son síncronas. Esto no supone un problema en ningún caso, pero no es conceptualmente correcto en situaciones en que la invocación de un servicio implica un largo tiempo de procesado. Ejemplos claros de este tipo de situaciones serían los servicios de impresión y escaneado de las IdD.

En éste tipo de situaciones, se optará por la recepción de la orden de trabajo, su preparación y una respuesta de Aceptado (código HTTP 202 [IETF1]) a la recepción de la tarea de forma síncrona. Toda comunicación posterior (inicio, estados intermedios, fin, etc.) se llevará a cabo mediante el uso de colas de mensajes. Esta práctica permite desacoplar cliente y servidor y limitar el número de conexiones activas entre piezas. Las comunicaciones asíncronas se representarán en los diagramas mediante líneas discontinuas.

5.4.1 Presencia y estado de los dispositivos

Como ya se ha establecido previamente, las IdD funcionan de forma autónoma. Pero para que se pueda acceder a los periféricos que disponibiliza, antes es necesario inventariar tanto las islas como los dispositivos.

Tal y como se muestra en la Ilustración 14, la gestión de presencia y estado de los periféricos se realiza informando a la NHC mediante el uso de colas de mensajes. A continuación se detallará la secuencia de este flujo.

- Bloque I: la ***IslaA*** arranca y publica un mensaje en el ***topicA*** informando de este hecho. El mensaje contiene el nombre de la isla, su dirección IP (***IPIslaA***) y el estado de encendido. Dicho mensaje se recibe en un procesador de mensajes. Éste aplica la máscara de red de las oficinas para obtener la dirección de red de la oficina, ***IPredIslaA***. Después, busca en la caché la correspondencia de esa dirección con un código de oficina, ***OficinaA***. Finalmente, guarda

en la caché la correspondencia entre el nombre de la isla, su oficina y su estado.

- Bloque II: una vez notificado su estado, la isla comienza a publicar un mensaje por cada dispositivo conectado. El procesador recibe dichos mensajes y almacena en la caché el nombre, tipo, estado y oficina asociada de cada uno de ellos.

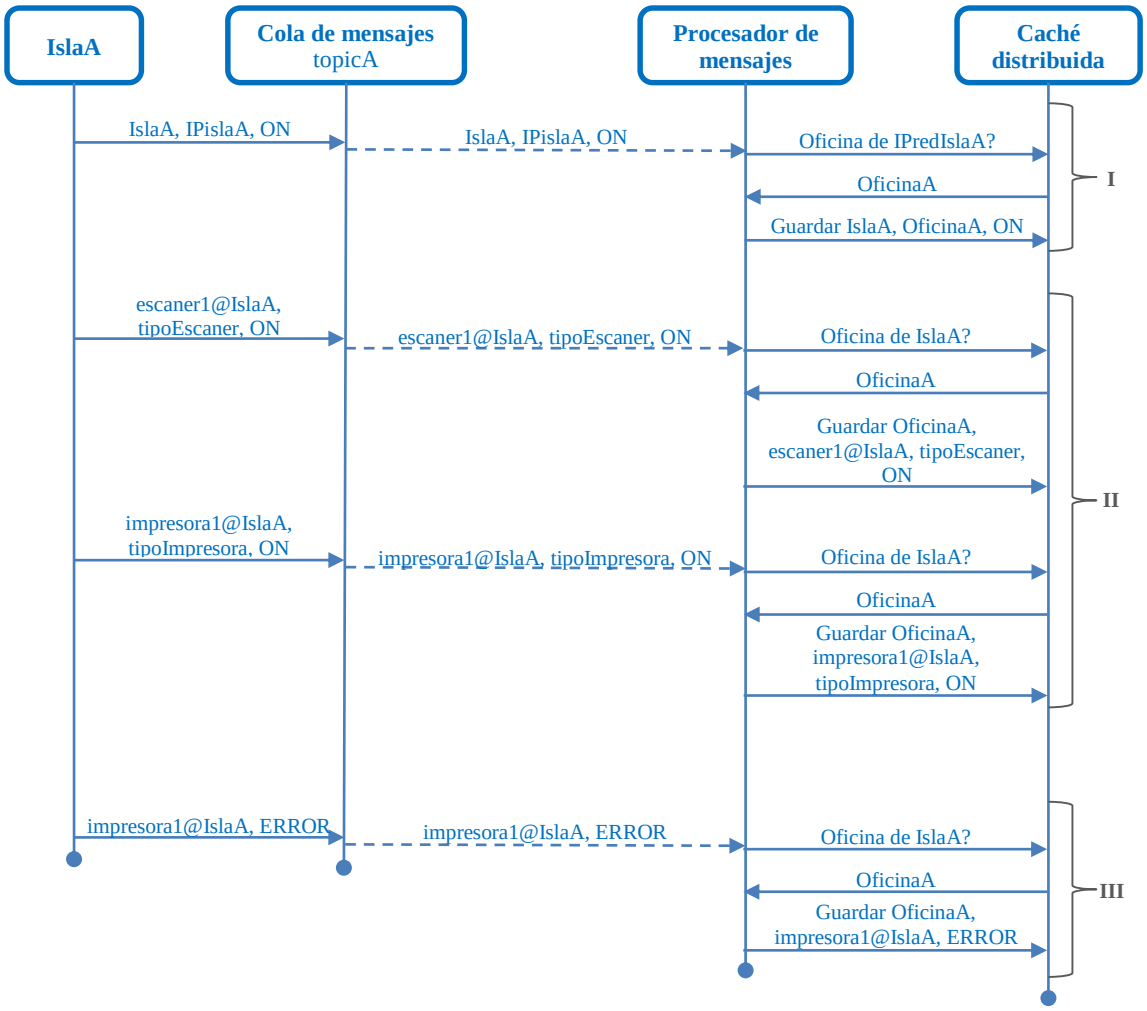

Ilustración 14 - Secuencia de presencia y estado de los dispositivos.

- Bloque III: posteriormente, la impresora de la ***IslaA*** sufre un fallo y se publica un mensaje informando del cambio de estado. El procesador recoge el mensaje y actualiza la caché a la nueva situación.

El resto de IdDs del escenario actuarían conforme a los bloques I y II. De esta forma, pasados unos minutos del arranque de las islas, la caché distribuida contendría un inventario completo de oficinas, islas, dispositivos y estados con sus respectivas relaciones.

5.4.2 Descubrimiento de dispositivos desde el EdT

El EdT necesita conocer una lista de dispositivos disponibles. La secuencia de descubrimiento de los mismos, mostrado en la Ilustración 15, se inicia con el arranque del EdT

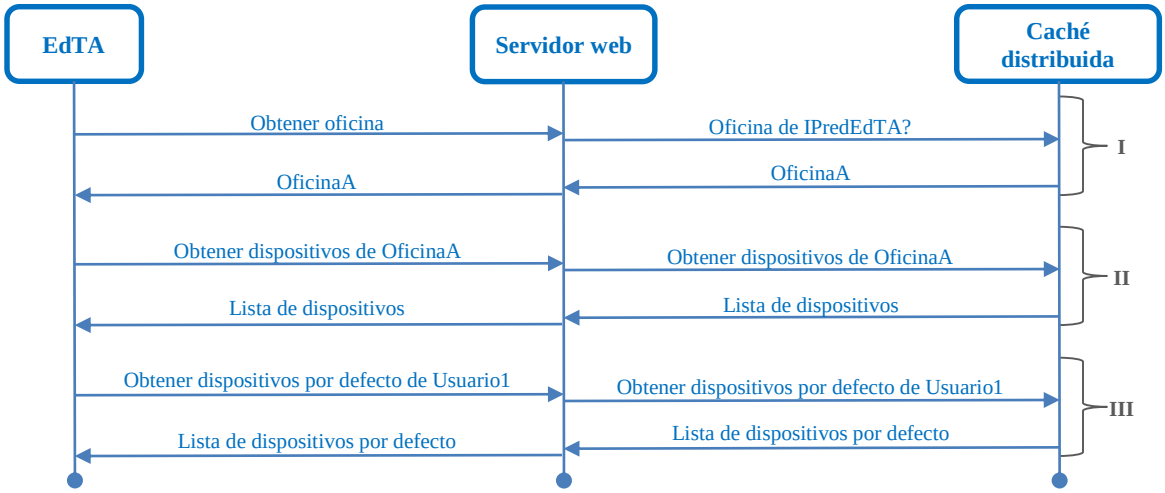

Ilustración 15 - Secuencia de descubrimiento de dispositivos desde el EdT.

- Bloque I: en primer lugar, el ***EdTA*** necesita ubicarse a sí mismo en una oficina. Para ello lanza una petición a un servicio de la NHC. El servicio obtiene la IP de ***EdTA*** y aplica la máscara de red de las oficinas para obtener la dirección de red de la oficina, ***IPredEdTA***, consulta la caché para obtener el código de oficina, ***OficinaA***, y se lo informa al ***EdTA*** en la respuesta.
- Bloque II: el ***EdTA*** solicita la lista de dispositivos presentes en la ***OficinaA***. El servicio que recoge la solicitud consulta la cache y le devuelve la lista. En nuestro caso concreto, la lista contendría los elementos de la Tabla 2.

Nombre de dispositivo	Tipo de dispositivo
escaner1@IslaA	tipoEscaner
impresora1@IslaA	tipoImpresora
impresora1@IslaB	tipoImpresora
escaner1@IslaC	tipoEscaner
impresora1@IslaC	tipoImpresora
escaner1@IslaD	tipoEscaner
impresora1@IslaD	tipoImpresora

Tabla 2 - Dispositivos presentes en la oficina.

- Bloque III: el ***EdTA*** solicita la lista de perfiles de dispositivos por defecto para el ***Usuario1***. El servicio que recoge la solicitud

consulta la cache y le devuelve la lista. En este caso, correspondería con los dispositivos de la Tabla 3.

Nombre del perfil	Tipo de dispositivo	Nombre de dispositivo
areaCliente	tipoEscaner	escaner1@IslaA
areaCliente	tipoImpresora	impresora1@IslaA
areaTrabajo	tipoEscaner	escaner1@IslaC
areaTrabajo	tipoImpresora	impresora1@IslaC
areaPrivada	tipoEscaner	escaner1@IslaD
areaPrivada	tipoImpresora	impresora1@IslaD

Tabla 3 - Perfiles de dispositivos del gestor.

Llegados a este punto, y en una situación real, el ***EdTA*** debería subscribirse a una cola dedicada para la oficina, ***topicOficinaA***, que recoge todos los eventos de dicha oficina. A través de esta cola se recibirán los subsiguientes cambios de estado en los periféricos. Sin embargo, obviaremos ese proceso en pro de una mayor claridad en las explicaciones siguientes.

5.4.3 Escaneo de un documento identificativo

En este punto, el cliente ha accedido a la contratación del producto. El gestor le ha solicitado un documento identificativo, por ejemplo el DNI, y lo ha introducido en el escáner por defecto del perfil activo. La Ilustración 16 muestra la secuencia de escaneado del documento. Ésta se compone básicamente de dos fases: escaneo y subida del documento resultante y procesado de reconocimiento para la extracción de datos relevantes. En este caso sólo es extraerá la foto del cliente para adjuntarla posteriormente a su perfil. Veamos más detalladamente la secuencia:

- Bloque I: el ***EdTA*** envía una solicitud de escaneo al ***IdDA*** que tiene el escáner seleccionado por defecto: ***escaner1@IslaA***. Recordemos que en este punto el ***IdDA*** pedirá confirmación de las credenciales del ***EdTA***. Una vez autenticadas las credenciales, se procede al escaneo y subida al gestor documental. La subida genera un identificador de documento, ***IdDoc1***, que es devuelto al ***EdTA***. Cada paso intermedio desde que se inicia el escaneo es informado al ***EdTA*** mediante sucesivos mensajes asíncronos a través de la cola ***topicOficinaA***.
- Bloque II: Ahora hay que procesar el documento obtenido. Como las tareas de reconocimiento son muy costosas en términos de tiempo y capacidad de cómputo, se procederá a iniciar el proceso previamente a la formalización del contrato. Para ello el ***EdTA*** coloca una tarea en el servidor ROC con un identificador de la tarea, ***IdJob1***, y el ***IdDoc1*** obtenido en la fase de escaneo. El

servidor de reconocimiento se descarga el fichero, lo procesa y obtiene la fotografía del mismo, ***Doc2***. Por último, lo sube al almacenamiento obteniendo ***IdDoc2*** y guarda en la caché la asociación entre trabajo y resultado para su posterior consulta.

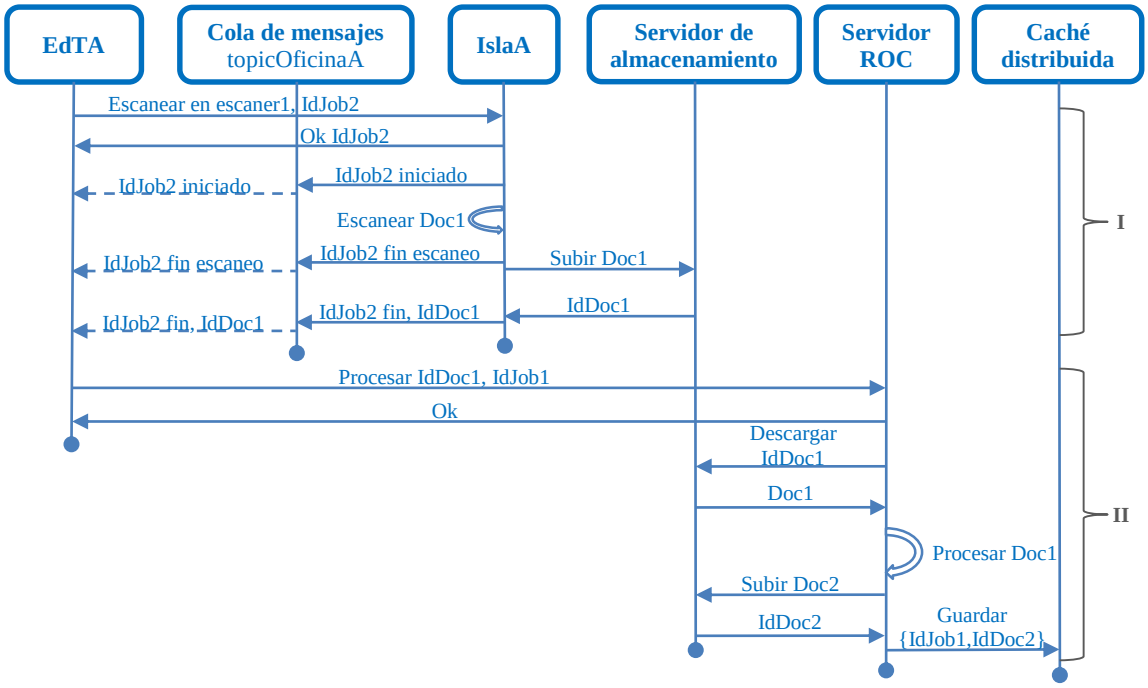

Ilustración 16 - Secuencia de escaneo de un documento identificativo.

5.4.4 Alta de cliente y contrato

Mientras el documento identificativo se está procesando en las instancias del NHC, el gestor rellena el formulario con datos del cliente, le solicita firmar en un campo del formulario (el elemento Canvas de HTML5 es idóneo para esta tarea) y procede a dar de alta al cliente y la contratación. La Ilustración 17 muestra la secuencia concreta que podríamos dividir en dos fases: por un lado la subida de la firma al gestor documental y, por otro, el alta de cliente y la contratación en sí mismas.

- Bloque I: el ***EdTA*** sube el documento con la firma del cliente al servidor de almacenamiento obteniendo el ***IdDoc3***.
- Bloque II: ahora ***EdTA*** solicita el alta de cliente y contrato adjuntando en la llamada todos los datos necesarios: código de producto (***Producto1***), datos del cliente (***DatosCliente***), identificadores del DNI y la firma (***IdDoc1*** e ***IdDoc3***) y el identificador del trabajo de extracción de la fotografía del DNI (***IdJob1***). El servidor genera un contrato (***Contrato1***) con los datos y firma del cliente y el tipo de producto. Lo almacena obteniendo el ***IdDoc4***. Después, recupera el ***IdDoc2*** de la caché. Y finalmente

da de alta el cliente y el contrato en la base de datos. Como resultado se genera un identificador de cliente, ***IdCliente1***. La respuesta devuelta al ***EdTA*** contiene los identificadores del cliente, y el contrato generado.

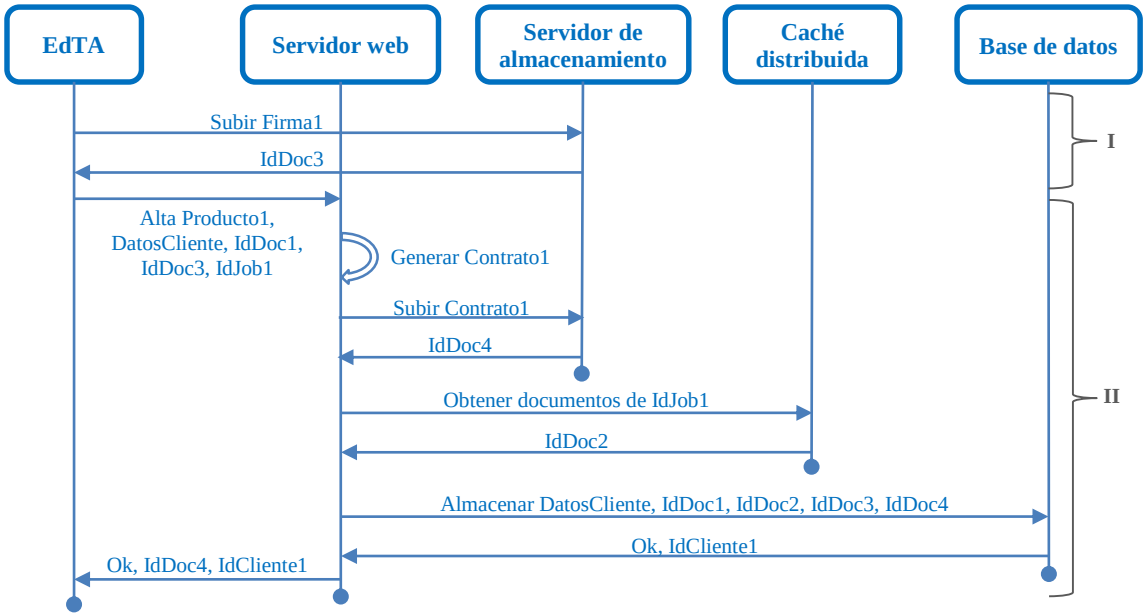

Ilustración 17 - Secuencia de alta de cliente y contrato.

5.4.5 Emisión de copias del contrato

Como se estableció al inicio en la explicación del caso de uso, se emitirán dos copias del contrato: una impresa para que el cliente se lleve en mano y otra enviada por correo electrónico.

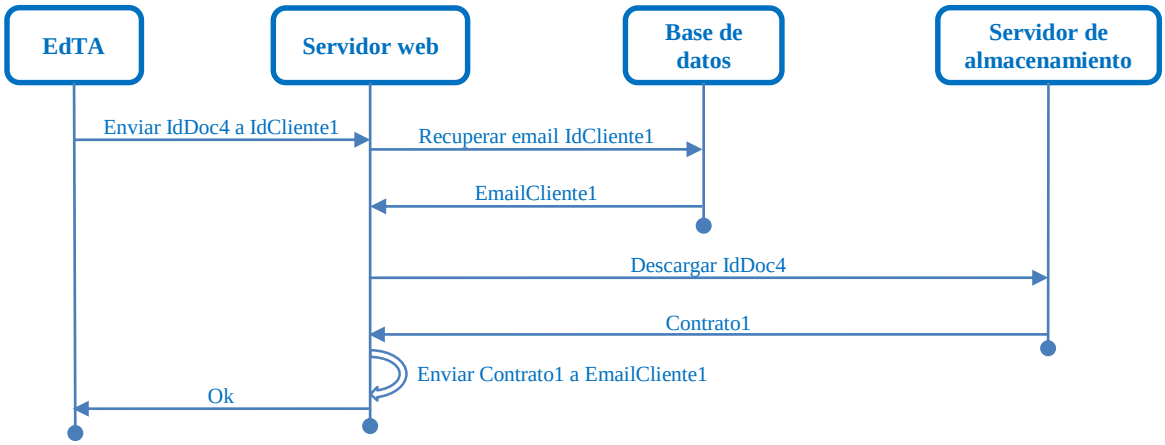

Ilustración 18 - Secuencia de envío del contrato.

La secuencia de envío del contrato se muestra en la Ilustración 18. En ella el EdTA solicita el envío del contrato, con ***IdDoc4***, al cliente, con ***IdCliente1***. El servicio que recoge

la petición, recupera la dirección de correo electrónico, ***EmailCliente1***, de la base de datos, se descarga el documento ***Contrato1***, compone el mensaje y lo envía.

La Ilustración 19, muestra la secuencia de impresión que, como recordaremos, incluye una situación de error en la impresora por defecto.

- Bloque I: el ***EdTA*** solicita un trabajo impresión de Contrato1, con ***IdDoc4***, en ***impresora1@IslaA***. Pero durante el inicio del proceso ocurre un error. Éste se informa en la respuesta al ***EdTA***.

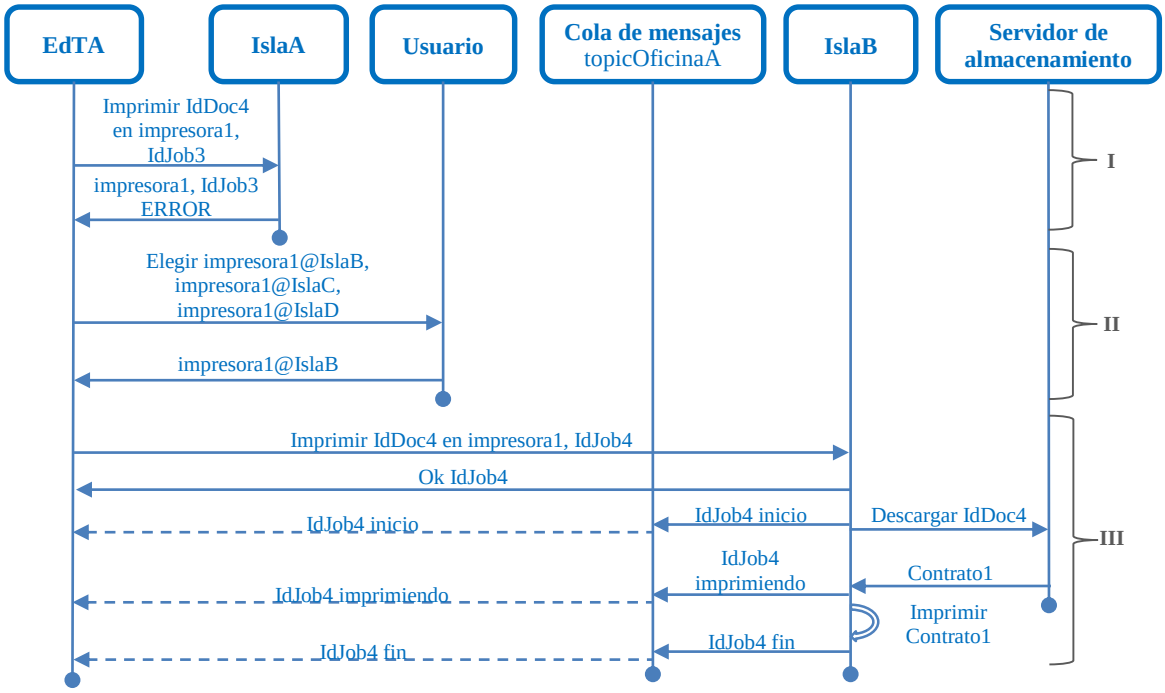

Ilustración 19 - Secuencia de impresión del contrato.

- Bloque II: ante la situación de indisponibilidad de la impresora por defecto, el ***EdTA*** pregunta al gestor cual desea utilizar de entre las demás disponibles. El gestor elige la otra impresora del área de cliente: ***impresora1@IslaB***.
- Bloque III: el ***EdTA*** solicita un trabajo de impresión del contrato en dicho dispositivo. ***IslaB***, al recibir la petición se descarga el documento con ***IdDoc4*** del gestor documental y lo imprime. Al igual que en la fase de escaneo, cada paso desde que se inicia la impresión es informado al ***EdTA*** mediante sucesivos mensajes asíncronos a través de la cola ***topicOficinaA***.

Capítulo 6

Conclusiones y líneas futuras

6.1 Conclusiones

El sector de la banca minorista se encuentra en un punto delicado tras la aparición de nuevos competidores, originarios principalmente del sector tecnológico. Éstos han traído consigo una innovadora forma de entender y ofrecer productos financieros que, cada día más, está calando entre la masa de clientes potenciales. Esta situación ha colocado a las entidades bancarias en un punto de inflexión. Por un lado están estas amenazas y por otro unas estructuras, modelos de negocio y plataformas tecnológicas desfasadas que impiden un giro ágil en el modelo de funcionamiento de dichas entidades.

Día tras día, los canales digitales ganan importancia en el volumen de negocio y hay quienes opinan que este hecho desembocará en el cierre masivo de las redes de oficinas de las entidades. Sin embargo, dicha red de oficinas aporta un valor intangible de cercanía y trato personal que difícilmente puede conseguirse por otros medios. El problema radica en que su estructura y concepción actual no satisfacen completamente las demandas de los usuarios. A lo largo de este proyecto se ha pretendido plantear una solución a esta situación.

Tras un análisis formal de la situación y diseño actuales de las oficinas, se llegó a la conclusión de que existen numerosas debilidades. Por un lado, existe poca capacidad de diferenciación en la disposición y uso del entorno de una oficina y no se confiere especial

relevancia al trato con el cliente. Por otro, las actuales plataformas tecnológicas que sirven a las oficinas son poco elásticas, tienen altos costes o presentan limitaciones para realizar implantaciones ágiles. Todo ello supone ciertos riesgos de funcionamiento y eficacia en el medio-largo plazo de cara a renovar los procesos de negocio o adoptar nuevas estrategias de interacción con el cliente.

Para solucionarlo se ha propuesto un paradigma de oficina en el que la cercanía y experiencia de usuario sean los ejes fundamentales del diseño. Se ha planteado una redistribución del espacio dando prioridad al área de cliente y a la forma en que cliente y gestor interaccionan. El uso de equipos portátiles 2 en 1 y un nuevo concepto de mobiliario, permite al cliente formar parte de las operaciones, que pueda seguir el proceso en primera persona y hablar con los empleados de tú a tú. Este modelo de oficina no elimina la demanda de servicios de caja ni la necesidad de uso de dispositivos. Para la primera, se plantea el uso de una renovada generación de autoservicios que permitan a los clientes llevar a cabo las operaciones del día a día como pago de impuestos, recibos y multas, ingreso de cheques o efectivo, etc.

Con el fin de solventar la segunda problemática, se ha presentado el concepto de Isla de Dispositivos. Una IdD es un elemento concentrador de dispositivos que funciona de forma autónoma y expone un conjunto de servicios que permitan utilizar la periferia de forma remota. Los servicios se pueden publicar mediante un Tomcat que, a su vez, hace uso de varias bibliotecas para acceder a los distintos periféricos. El uso de estas islas permite reducir costes y aumentar el tiempo de disponibilidad de los dispositivos en su conjunto. Lo que a su vez redunda en un aumento de la productividad. Además, las IdDs se podrán identificar por colores mediante el uso de dispositivos luminosos RGB.

Asimismo, se han esbozado algunas mejoras en lo que a prácticas y servicios de banca se refiere. Estas propuestas se pueden dividir, por un lado, en aquellas que afectan directamente a las oficinas como adaptar los horarios de apertura o desligar a los clientes de una oficina concreta. Y, por otra parte, también se han planteado mejoras que afecta a la entidad en su conjunto como la racionalización del catálogo de productos, la integración con productos de terceros de uso cotidiano o la aplicación de técnicas de Big Data para conseguir un mayor conocimiento de las demandas y necesidades de la base de clientes.

Sin embargo, es complicado emprender cualquiera de estas mejoras sin antes afrontar un cambio en la arquitectura tecnológica que soporta el modelo productivo. Para hacer frente a este problema se ha propuesto una arquitectura global y de oficina construida sobre dos pilares básicos: el uso de una arquitectura de servicios y de elementos de computación en la nube.

En cuanto a la arquitectura de servicios, los beneficios de su uso permiten dinamizar la implantación de nuevos productos y servicios. Se han barajado dos opciones: SOAP y REST. Tras analizar los pros y los contras de cada una de ellas, se ha optado por una solución híbrida. La primera se emplea en los procesos internos de la entidad y todos

aquellos externos que requieran de un añadido de seguridad. Y la segunda se utiliza para todo lo demás, especialmente en el Escritorio de Trabajo del gestor y el acceso desde los portales de cliente. Esta solución de compromiso permite aunar el dinamismo, la versatilidad y la ligereza de REST con la seguridad, transaccionalidad y el concepto de contrato y estado de SOAP.

Respecto a la utilización de elementos de computación en la nube, las principales ventajas que aporta frente al modelo anterior son: escalabilidad y elasticidad auto-aprovisionados y compartición de infraestructura. Estos dos puntos permitirán afrontar, a menor coste, el despliegue y mantenimiento de servicios conservando un alto rendimiento y disponibilidad. Se han examinado los distintos tipos de servicios que existen actualmente y estudiado numerosos productos comerciales. Tras este análisis y, dado que una empresa del sector financiero maneja información inherentemente delicada, se propone la creación de una nube híbrida corporativa. Bajo este esquema, todos los procesos e información sensible quedan confinados en la parte privada de la nube donde la entidad tendrá control total sobre ellos. El resto de procesos e información se pueden ubicar en los servicios de un proveedor externo. Siempre y cuando éste almacene los datos en servidores en la UE o que, como mínimo, cumpla con los principios Safe Harbour.

Como proveedores de la parte pública se han elegido Amazon y Google pues ambos tienen la certificación Safe Harbour, disponen de CPDs tanto en Europa como a escala global, tienen cifras de fiabilidad muy elevadas. Adicionalmente, Google dispone de herramientas en la nube de uso común para los empleados como son Gmail, Drive y Docs.

Para aprovechar las características que ofrece la arquitectura propuesta, y de cara a mejorar la experiencia de usuario en el nuevo modelo de interacción cliente-gestor, se plantea un rediseño del Escritorio de Trabajo. Éste se configura como una aplicación web, concretamente una RIA, construida en HTML5, CSS3 y Javascript. Esta decisión de diseño permite delegar el despliegue en los servidores web, distribuir la carga de trabajo entre los clientes y los servidores y conseguir interfaces de usuario más ricas con un bajo consumo de ancho de banda.

Dado que las oficinas son puntos de acceso público y que la solución implica el uso de una red WiFi, ha sido conveniente establecer unos mecanismos de seguridad. En primer lugar, la red inalámbrica está cifrada; así como la VPN que comunica la oficina con la nube. Además se demanda el uso de HTTPS en todas las comunicaciones. Como añadido, se ha establecido un protocolo destinado a autenticar y autorizar el equipo desde el que se accede a los servicios. Para ello se obtiene un identificador del navegador mediante Browser Fingerprinting y se autoriza su registro mediante una OTP al gestor.

6.2 Líneas futuras

Por último, sólo resta por hablar de las posibles líneas de trabajo que quedan abiertas tras este documento. En lo que respecta a la modernización tecnológica, todas las entidades cuentan, en mayor o menor medida, con sistemas legacy. Éstos son sistemas antiguos o propietarios que, por un motivo u otro, deben seguir prestando servicio. No se ha tratado en el proyecto por ser una labor muy extensa e individualizada para cada sistema. Sin embargo, sería necesario realizar un estudio pormenorizado que analizase la posibilidad de eliminarlos o, cuanto menos, integrarlos de forma adecuada en la arquitectura propuesta.

La arquitectura propuesta se ha planteado de forma que pueda soportar nuevos tipos de productos y formas de interacción con el cliente. Por lo que otra línea de trabajo consistiría en abordar bien la creación de algunos de los servicios propuestos, bien explorar soluciones o productos innovadores que permitan a las entidades obtener un valor diferenciador respecto a la competencia. Por ejemplo, el uso de NFC o códigos QR como sistema de identificación de un cliente o para pagos peer-to-peer, plataformas de crowdfunding, una aplicación para pagos compartidos, otra para la gestión remota de la cuenta de gastos de un menor, asesoría y servicios financieros a domicilio, etc. Cualquier innovación, por pequeña que sea y tenga o no un beneficio económico directo, puede suponer un incremento en la tasa de captación y retención de clientes. El objetivo último es ganarse la confianza del usuario y convertirse en una plataforma moderna, digital y ubicua de uso diario.

Otro punto de estudio muy interesante consistiría en estudiar la posibilidad de crear y ofrecer una API mediante la cual otras empresas, ajenas al sector, puedan integrar en sus productos ciertas capacidades de servicios financieros.

Una vía adicional de estudio consistiría en analizar el coste de implantación del modelo propuesto en términos de costes directos, indirectos y de oportunidad. Es posible que pequeñas entidades no puedan afrontar el gasto de la transformación. Pero en tal caso, es posible que se viesen obligadas a cesar su actividad en el medio-largo plazo, por lo que deberían barajar la opción de transformar su negocio en servicios y productos de nicho.

Glosario

API

O interfaz de programación de aplicaciones en español, es una capa de abstracción compuesta por un conjunto de subrutinas, funciones, métodos o procedimientos que ofrece cierta biblioteca para ser utilizada por otro software.

Arquitectura cliente-servidor

Es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes.

Autenticación

Es un concepto de seguridad tiene como objetivo confirmar la veracidad de una afirmación. En los escenarios tratados se utiliza para identificar de forma segura y unívoca a uno o ambos extremos de la comunicación.

Autorización

Es un concepto de seguridad que extiende al de autenticación en tanto en cuanto tiene como objetivo adicional confirmar que el peticionario es quién dice ser y que tiene permiso para acceder al recurso o servicio solicitado.

Autoservicios

Son dispositivos pesados que posibilitan que el cliente efectúe operativas básicas, posiblemente fuera de horario comercial y sin supervisión de un empleado.

Big Data

Se refiere a un conjunto de técnicas de recogida, filtrado, ordenación y análisis de datos de diferentes tipos y fuentes que, a priori, podrían parecer no relacionados.

Canal ICA

Es un protocolo utilizado en entornos virtualizados permite la creación de un canal virtual que puede transportar, entre otros, las señales de bajo nivel de los periféricos.

Computación en la nube

Es un paradigma que permite ofrecer servicios de computación a través de una red.

CPD

O centro de procesamiento de datos, es aquella ubicación donde se concentran los recursos necesarios para el procesamiento de la información de una organización.

Dispensador de divisa

Es una máquina similar a un reciclador pero carece de la capacidad para reconocer y validar la divisa. Por lo que no permite ingresar efectivo.

Freemium

Es un esquema de tarificación en el que existe un nivel de servicio de uso gratuito pero limitado en tiempo o funcionalidad. Si el cliente está interesado en el producto tiene la opción de subir a uno o más niveles de servicio, en los que dispone de mayores capacidades, mediante una subscripción de pago.

HTTP

Es un protocolo de aplicación utilizado eminentemente en los entornos web.

HTTPS

Es un protocolo basado en HTTPS que permite cifrar las comunicaciones extremo a extremo mediante el uso de TLS.

JSON

Es un formato ligero de intercambio de datos.

NFC

O comunicación de campo cercano en español, es una tecnología de comunicación inalámbrica, de corto alcance y alta frecuencia que permite el intercambio de datos entre dispositivos.

OAUTH

Es un protocolo que permite flujos simples de autorización para sitios web o aplicaciones informáticas.

OTP

O contraseña de un solo uso en español, es una contraseña válida solo para una autenticación. Su envío se suele realizar por un canal secundario al de la autenticación principal.

Reciclador de divisa

Es un dispositivo acorazado que permite ingresar y dispensar billetes de una o más divisas directamente desde un puesto de oficina.

RPC

Es un protocolo que permite a una entidad ejecutar código de otra de forma remota.

SAML

Es un estándar abierto que define un esquema XML para el intercambio de datos de autenticación y autorización

SMTP

Es un protocolo de aplicación utilizado para la transferencia de correos electrónicos.

SOAP

Es un protocolo de mensajería basado en XML utilizado para comunicar Web services.

TCP

Es un protocolo de transporte fiable que se utiliza en la transmisión de datos.

TLS

Es un protocolo criptográfico que posibilita el cifrado de comunicaciones.

Transacción ACID

Así se denominan a las transacciones que cumplan con unos ciertos parámetros de atomicidad, consistencia, aislamiento y durabilidad.

Transacciones peer-to-peer

Es un recurso financiero por el que dos individuos pueden intercambiar dinero de forma remota.

VPLS

Es un tipo de VPN que permite conectividad punto a punto a través de túneles de nivel 2.

VPN

Es una tecnología de red que permite una extensión segura de una LAN sobre una red pública o no controlada como Internet.

WADL

Es un lenguaje de definición de servicios REST.

WSDL

Es un lenguaje de definición de Web Services basado en XML.

XML

Es un lenguaje descriptivo basado en etiquetas que es legible tanto por máquinas como humanos.

Referencias

Referencias bibliográficas

[ACC1]. Accenture Research. *2014 North America Consumer Digital Banking Survey. The Digital Disruption in Banking. Demons, demands, and dividends*. En: www.accenture.com [en línea]. Disponible en: https://www.accenture.com/in-en/~/media/Accenture/Conversion-Assets/DotCom/Documents/Global/PDF/Industries_5/Accenture-2014-NA-Consumer-Digital-Banking-Survey.pdf

[ACC2]. Accenture Research, *Banking 2020 Thought Leadership Series A Critical Balancing Act: US Retail Banking in the Digital Era*. En: www.accenture.com [en línea]. Disponible en: http://nstore.accenture.com/IM/FinancialServices/AccentureLibrary/data/pdf/US_Retail_Banking_in_the_Digital_Era.pdf

[BBVA1]. ALONSO, Rebeca, 19 Noviembre 2012. *Ir al banco ya no será como antes*. En: e-volucion.elnortedecastilla.es [en línea]. Disponible en: http://e-volucion.elnortedecastilla.es/negocio-digital/ir-al-banco-ya-no-sera-como-antes-19112012.html [consulta: 31 Agosto 2015]

[BIG1]. DUFFY, Anthony. *Unlocking the potential of Big Data*. En: www.bankingtech.com [en línea]. Disponible en: http://www.bankingtech.com/58812/unlocking-the-potential-of-big-data/ [consulta: 31 Agosto 2015]

[CCUT1]. BUTLER, Brandon, 12 Enero 2015. *Which cloud providers had the best uptime last year?* En: www.networkworld.com [en línea]. Disponible en:

http://www.networkworld.com/article/2866950/cloud-computing/which-cloud-providers-had-the-best-uptime-last-year.html [consulta: 31 Agosto 2015]

[CCUT2]. JUDGE, Peter, 14 Enero 2015. *Microsoft Azure trails in cloud reliability* En: www.datacenterdynamics.com [en línea]. Disponible en: http://www.datacenterdynamics.com/app-cloud/microsoft-azure-trails-in-cloud-reliability/93097.fullarticle [consulta: 31 Agosto 2015]

[CHR1]. JANC, Artur y ZALEWSKI, Michal. *Technical analysis of client identification mechanisms*. En: www.chromium.org [en línea]. Disponible en: https://www.chromium.org/Home/chromium-security/client-identification-mechanisms [consulta: 31 Agosto 2015]

[EZIN1]. CHOPRA, Rohit d., 24 Abril 2010. *Rich Internet Application Advantages*. En: ezinearticles.com [en línea]. Disponible en: http://ezinearticles.com/?Rich-Internet-Application-Advantages&id=4169336 [consulta: 31 Agosto 2015]

[FORB1]. COLUMBUS, Louis, 24 Enero 2015. *Roundup Of Cloud Computing Forecasts And Market Estimates, 2015*. En: www.forbes.com [en línea]. Disponible en: http://www.forbes.com/sites/louiscolumbus/2015/01/24/roundup-of-cloud-computing-forecasts-and-market-estimates-2015/ [consulta: 31 Agosto 2015]

[GAR1]. TSIDULKO, Joseph, 20 Mayo 2015. *Here's Who Made Gartner's 2015 Cloud IaaS Magic Quadrant*. En: www.crn.com [en línea]. Disponible en: http://www.crn.com/slide-shows/cloud/300076877/heres-who-made-gartners-2015-cloud-iaas-magic-quadrant.htm [consulta: 31 Agosto 2015]

[GAR2]. VERGE, Jason, 28 Mayo 2015. *Gartner: AWS Pulls Further Ahead of Others in IaaS Cloud Market*. En: www.datacenterknowledge.com [en línea]. Disponible en: http://www.datacenterknowledge.com/archives/2015/05/28/gartner-aws-pulls-further-ahead-in-iaas-cloud-market/ [consulta: 31 Agosto 2015]

[HBR1]. BUSCH, Wayne y MORENO, Juan Pedro, 20 Febrero 2014. *Banks' New Competitors: Starbucks, Google, and Alibaba*. En: www.hbr.com [en línea]. Disponible en: https://hbr.org/2014/02/banks-new-competitors-starbucks-google-and-alibaba/ [consulta: 31 Agosto 2015]

[IBM1]. *Deriving Business Insight from Big Data in Banking*. En: www.ibm.com [en línea]. Disponible en: http://www-01.ibm.com/software/data/bigdata/industry-banking.html [consulta: 31 Agosto 2015]

[ICA1]. 18 Mayo 2015. *Citrix ICA Virtual Channels Overview*. En: support.citrix.com [en línea]. Disponible en: https://support.citrix.com/article/CTX116890 [consulta: 31 Agosto 2015]

[ICA2]. VERMEULEN, Stefan, 07 Diciembre 2005. *Dealing with Poor Bandwidth and Latency*. En: www.virtualizationadmin.com [en línea]. Disponible en: http://www.virtualizationadmin.com/articles-tutorials/terminal-services/performance/poor-bandwidth-latency.html [consulta: 31 Agosto 2015]

[IETF1]. Fielding R., Gettys J., Mogul J., Frystyk H., Masinter L., Leach P. y Berners-Lee T., "*Hypertext Transfer Protocol -- HTTP/1.1"*, Internet Engineering Task Force, Junio 1999. Disponible en: https://www.ietf.org/rfc/rfc2616.txt

[JSON1]. PRAGMATEEK, 10 Junio 2013. *JSON vs. XML: Some hard numbers about verbosity*. En: www.codeproject.com [en línea]. Disponible en: http://www.codeproject.com/Articles/604720/JSON-vs-XML-Some-hard-numbers-about-verbosity [consulta: 31 Agosto 2015]

[JSRIA1]. BORLINGHAUS, Tobi y CLASE, Daniel, 5 Marzo 2015. *Comparison of 4 popular JavaScript MV* frameworks (part 2)*. En: www.developereconomics.com [en línea]. Disponible en: http://www.developereconomics.com/comparison-4-popular-javascript-mv-frameworks-part-2/ [consulta: 31 Agosto 2015]

[JSRIA2]. ARORA, Sunil, 21 Febrero 2015. *JavaScript Frameworks: The Best 10 for Modern Web Apps*. En: noeticforce.com [en línea]. Disponible en: http://noeticforce.com/best-Javascript-frameworks-for-single-page-modern-web-applications [consulta: 31 Agosto 2015]

[Kurt]. SALMON, Kurt. Julio 2013, *Phygital and other digital challenges for retail banks*. En: www.kurtsalmon.com [en línea]. Disponible en: http://www.kurtsalmon.com/uploads/Efma-Report-Phygital-Kurt-Salmon.pdf

[MOZ1]. *HTML5*. En: developer.mozilla.org [en línea]. Disponible en: https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5 [consulta: 31 Agosto 2015]

[MOZ2]. ROUGET, Paul y DORME, Geoffrey. *HTML5 & Friends*. En: developer.mozilla.org [en línea]. Disponible en: https://developer.cdn.mozilla.net/media/uploads/demos/p/a/paulrouget/html5-dashboard/demo_package/index.html [consulta: 31 Agosto 2015]

[MSDN1]. SKONNARD, Aaron, Octubre 2003. *Understanding WSDL*. En: msdn.microsoft.com [en línea]. Disponible en: https://msdn.microsoft.com/en-us/library/ms996486.aspx [consulta: 31 Agosto 2015]

[NIST1]. MELL, Peter y GRANCE, Timothy, "*The NIST Definition of Cloud Computing"*, National Institute of Standards and Technology, Septiembre 2011. Disponible en: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf [consulta: 31 Agosto 2015]

[OAS1]. L. Clement, A. Hately, C. von Riegen y T. Rogers, "*UDDI Version 3.0.2"*, Organization for the Advancement of Structured Information Standards, 2004. Disponible en: https://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm

[OAUTH1]. OH, Changhun, 4 Septiembre 2012. *Dancing with OAuth: Understanding how Authorization Works*. En: www.cubrid.org [en línea]. Disponible en: http://www.cubrid.org/blog/dev-platform/dancing-with-oauth-understanding-how-authorization-works/ [consulta: 31 Agosto 2015]

[PAN1]. ECKERSLEY, Peter. *How Unique Is Your Web Browser?* En: panopticlick.eff.org [en línea]. Disponible en: https://panopticlick.eff.org/browser-uniqueness.pdf

[PWC1]. SHIPMAN, John, VERMEULEN, Otto, SNG, Jimmy, YIM Sang-Pyo, SULLIVAN, Robert P. y VELASCO, Francisco. *Eyes wide shut. Global insights and actions for banks in the digital age*. En: www.pwc.com [en línea]. Disponible en: https://www.pwc.com/im/en/publications/assets/banking/global_digital_banking_survey1.pdf

[SHC1]. "*The application of Commission Decision on the adequate protection of personal data provided by the Safe Harbor Privacy Principles (2002)"*, COMMISSION OF THE EUROPEAN COMMUNITIES, 13 Febrero 2002. Disponible en: http://web.archive.org/web/20060724174359/http://www.ec.europa.eu/justice_home/fsj/privacy/docs/adequacy/sec-2002-196/sec-2002-196_en.pdf

[SHC2]. "*The implementation of Commission Decision on the adequate protection of personal data provided by the Safe Harbor Privacy Principles (2004)"* COMMISSION OF THE EUROPEAN COMMUNITIES, 20 Noviembre 2004. Disponible en: http://web.archive.org/web/20060724173657/http://www.ec.europa.eu/justice_home/fsj/privacy/docs/adequacy/sec-2004-1323_en.pdf

[SHC3]. CONNOLLY, Chris, 02 Diciembre 2008. *US Safe Harbor - Fact or Fiction? (2008)*. En: www.galexia.com [en línea]. Disponible en: http://www.galexia.com/public/research/assets/safe_harbor_fact_or_fiction_2008 [consulta: 31 Agosto 2015]

[SvR1]. CHAPPELL, David, *SOAP vs. REST: Complements or Competitors?* En: www.esri.com [en línea]. Disponible en: http://downloads2.esri.com/campus/uploads/library/pdfs/98689.pdf

[SvR2]. FRANCIA, Steve, 15 Enero 2010. *REST Vs SOAP, The Difference Between Soap And Rest*. En: spf13.com [en línea]. Disponible en: http://spf13.com/post/soap-vs-rest [consulta: 31 Agosto 2015]

[TECHR1]. MATTESON, Scott, 8 Agosto 2013. *10 things you shouldn't virtualize*. En: www.techrepublic.com [en línea]. Disponible en: http://www.techrepublic.com/blog/10-things/10-things-you-shouldnt-virtualize/ [consulta: 31 Agosto 2015]

[TOM1]. MAPLE, Simon, 20 Junio 2013. *The Great Java Application Server Debate*. En: www.wired.com [en línea]. Disponible en: http://insights.wired.com/profiles/blogs/the-great-java-application-server-debate [consulta: 31 Agosto 2015]

[UoM1]. TOOSI, Adel Nadjaran, CALHEIROS Rodrigo N. y BUYYA, Rajkumar, "*Interconnected Cloud Computing Environments: Challenges, Taxonomy and Survey"*, The University of Melbourne, 2013. Disponible en: http://www.cloudbus.org/papers/InterCloud-ACMCS.pdf

[W3O1]. D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte y D. Winer, "*Simple Object Access Protocol (SOAP) 1.1"*, World Wide Web Consortium, 2000. Disponible en: http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[W3O2]. E. Christensen, F. Curbera, G. Meredith y S. Weerawarana, "*Web Services Description Language (WSDL) 1.1"*, World Wide Web Consortium, 2001. Disponible en: http://www.w3.org/TR/wsdl

[W3O3]. BOS, Bert, 2012. *HTML5 & the Open Web Platform*. En: www.w3.org [en línea]. Disponible en: http://www.w3.org/Talks/2012/0125-HTML-Tehran/ [consulta: 31 Agosto 2015]

[W3S1]. *SOAP Introduction*. En: www.w3schools.com [en línea]. Disponible en: http://www.w3schools.com/webservices/ws_soap_intro.asp [consulta: 31 Agosto 2015]

[WADL1]. Hadley, Marc, *Web Application Description Language (WADL)*, World Wide Web Consortium, 2009 [en línea]. Disponible en: http://www.w3.org/Submission/2009/SUBM-wadl-20090831/

Referencias enciclopédicas

[WIKI01]. *Big Data*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Big_data [consulta: 31 Agosto 2015]

[WIKI02]. *Service-Oriented Architecture*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Service-oriented_architecture [consulta: 31 Agosto 2015]

[WIKI03]. *SOAP*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/SOAP [consulta: 31 Agosto 2015]

[WIKI04]. *Web Services Description Language*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Web_Services_Description_Language [consulta: 31 Agosto 2015]

[WIKI05]. *Universal Description Discovery and Integration*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Universal_Description_Discovery_and_Integration [consulta: 31 Agosto 2015]

[WIKI06]. *Representational state transfer*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Representational_state_transfer [consulta: 31 Agosto 2015]

[WIK07]. *JSON*. En: www.wikipedia.org [en línea]. Disponible en: http://es.wikipedia.org/wiki/JSON [consulta: 31 Agosto 2015]

[WIK08]. *Web Application Description Language*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Web_Application_Description_Language [consulta: 31 Agosto 2015]

[WIK09]. *Authentication*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Authentication [consulta: 31 Agosto 2015]

[WIKI10]. *Authorization (computer access control)*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Authorization_%28computer_access_control%29 [consulta: 31 Agosto 2015]

[WIKI11]. *HTTPS*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/HTTPS [consulta: 31 Agosto 2015]

[WIKI12]. *Basic access authentication*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Basic_access_authentication [consulta: 31 Agosto 2015]

[WIKI13]. *Security Assertion Markup Language*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Security_Assertion_Markup_Language [consulta: 31 Agosto 2015]

[WIKI14]. *OAuth*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/OAuth [consulta: 31 Agosto 2015]

[WIKI15]. *Cloud Computing*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Cloud_computing [consulta: 31 Agosto 2015]

[WIKI16]. *Multitenancy*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Multitenancy [consulta: 31 Agosto 2015]

[WIKI17]. *Computación en la nube*. En: www.wikimedia.org [en línea]. Disponible en: https://upload.wikimedia.org/wikipedia/commons/f/ff/Cloud_computing-es.svg [consulta: 31 Agosto 2015]

[WIKI18]. *Software as a service*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Software_as_a_service [consulta: 31 Agosto 2015]

[WIKI19]. *Platform as a service*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Platform_as_a_service [consulta: 31 Agosto 2015]

[WIKI20]. *Infrastructure as a service*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Cloud_computing#Infrastructure_as_a_service_.28IaaS.29 [consulta: 31 Agosto 2015]

[WIKI21]. *Distributed cloud*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Cloud_computing#Distributed_cloud [consulta: 31 Agosto 2015]

[WIKI22]. *Intercloud*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Intercloud [consulta: 31 Agosto 2015]

[WIKI23]. *Multicloud*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Multicloud [consulta: 31 Agosto 2015]

[WIKI24]. *Cloud computing issues*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Cloud_computing_issues [consulta: 31 Agosto 2015]

[WIKI25]. *Freemium*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Freemium [consulta: 31 Agosto 2015]

[WIKI26]. *Hypervisor*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Hypervisor [consulta: 31 Agosto 2015]

[WIKI27]. *Rich Internet Application*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Rich_Internet_application [consulta: 31 Agosto 2015]

[WIKI28]. *Model–view–controller*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Model-view-controller [consulta: 31 Agosto 2015]

[WIKI29]. *Model View ViewModel*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Model_View_ViewModel [consulta: 31 Agosto 2015]

[WIKI30]. *Windows Open Services Architecture*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Windows_Open_Services_Architecture [consulta: 31 Agosto 2015]

[WIKI31]. *CEN/XFS*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/CEN/XFS [consulta: 31 Agosto 2015]

[WIKI32]. *JavaPOS*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/JavaPOS [consulta: 31 Agosto 2015]

[WIKI33]. *J/XFS*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/J/XFS [consulta: 31 Agosto 2015]

[WIKI34]. *Xpeak*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Xpeak [consulta: 31 Agosto 2015]

[WIKI35]. *Java Native Interface*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Java_Native_Interface [consulta: 31 Agosto 2015]

[WIKI36]. *Man-in-the-middle attack*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Man-in-the-middle_attack [consulta: 31 Agosto 2015]

[WIKI37]. *Optical character recognition*. En: www.wikipedia.org [en línea]. Disponible en: https://en.wikipedia.org/wiki/Optical_character_recognition [consulta: 31 Agosto 2015]

Referencias a servicios, productos y bibliotecas

[pACS01]. *Apache CloudStack*. En: www.apache.org [en línea]. Disponible en: https://cloudstack.apache.org [consulta: 31 Agosto 2015]

[pANGJS]. *AngularJS*. En: angularjs.org [en línea]. Disponible en: https://angularjs.org/ [consulta: 31 Agosto 2015]

[pAWS01]. *Amazon Web Services*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com [consulta: 31 Agosto 2015]

[pAWS02]. *Amazon Elastic Beanstalk*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/elasticbeanstalk [consulta: 31 Agosto 2015]

[pAWS03]. *Amazon Relational Database Service*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/rds [consulta: 31 Agosto 2015]

[pAWS04]. *Amazon DynamoDB*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/dynamodb [consulta: 31 Agosto 2015]

[pAWS05]. *Amazon Redshift*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/redshift [consulta: 31 Agosto 2015]

[pAWS06]. *Amazon ElastiCache*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/elasticache [consulta: 31 Agosto 2015]

[pAWS07]. *Amazon Simple Storage Service*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/s3 [consulta: 31 Agosto 2015]

[pAWS08]. *Amazon Simple Queue Service*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/sqs [consulta: 31 Agosto 2015]

[pAWS09]. *Amazon Elastic MapReduce*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/elasticmapreduce [consulta: 31 Agosto 2015]

[pAWS10]. *Amazon Elastic Compute Cloud*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/ec2 [consulta: 31 Agosto 2015]

[pAWS11]. *Amazon Elastic Load Balancing*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/elasticloadbalancing [consulta: 31 Agosto 2015]

[pAWS12]. *Amazon Route 53*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/route53 [consulta: 31 Agosto 2015]

[pAWS13]. *Amazon Virtual Private Cloud*. En: aws.amazon.com [en línea]. Disponible en: https://aws.amazon.com/vpc [consulta: 31 Agosto 2015]

[pBLINK]. *Blink(1)*. En: thingm.com [en línea]. Disponible en: https://blink1.thingm.com [consulta: 31 Agosto 2015]

[pCTX01]. *Citrix CloudPlatform*. En: www.citrix.com [en línea]. Disponible en: https://www.citrix.com/products/cloudplatform/overview.html [consulta: 31 Agosto 2015]

[pDEV1]. *TWAIN*. En: www.twain.org [en línea]. Disponible en: http://www.twain.org [consulta: 31 Agosto 2015]

[pDEV2]. *Scanner Access Now Easy*. En: www.sane-project.org [en línea]. Disponible en:

http://www.sane-project.org [consulta: 31 Agosto 2015]

[pDEV3]. *ISIS Document Imaging*. En: www.emc.com [en línea]. Disponible en: https://community.emc.com/community/edn/isis [consulta: 31 Agosto 2015]

[pDEV4]. *Windows Image Acquisition (WIA)*. En: msdn.microsoft.com [en línea]. Disponible en: https://msdn.microsoft.com/en-us/library/ms630368(VS.85).aspx [consulta: 31 Agosto 2015]

[pGCP01]. *Google Cloud Platform*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/ [consulta: 31 Agosto 2015]

[pGCP02]. *Google App Engine*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/appengine [consulta: 31 Agosto 2015]

[pGCP03]. *Google Cloud SQL*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/sql [consulta: 31 Agosto 2015]

[pGCP04]. *Google Cloud Datastore*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/datastore [consulta: 31 Agosto 2015]

[pGCP05]. *Google Cloud Bigtable*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/bigtable [consulta: 31 Agosto 2015]

[pGCP06]. *Google Cloud Storage*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/storage [consulta: 31 Agosto 2015]

[pGCP07]. *Google Cloud Pub/Sub*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/pubsub [consulta: 31 Agosto 2015]

[pGCP08]. *Google BigQuery*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/bigquery [consulta: 31 Agosto 2015]

[pGCP09]. *Google Compute Engine*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/container-engine [consulta: 31 Agosto 2015]

[pGCP10]. *Google Load Balancing*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/networking/#loadbalancing [consulta: 31 Agosto 2015]

[pGCP11]. *Google Cloud DNS*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/dns [consulta: 31 Agosto 2015]

[pGCP12]. *Google Interconnect*. En: cloud.google.com [en línea]. Disponible en: https://cloud.google.com/interconnect [consulta: 31 Agosto 2015]

[pJCon1]. *Jetty*. En: www.eclipse.org [en línea]. Disponible en: http://www.eclipse.org/jetty [consulta: 31 Agosto 2015]

[pJCon2]. *Apache Tomcat*. En: www.apache.org [en línea]. Disponible en: http://tomcat.apache.org [consulta: 31 Agosto 2015]

[pJCon3]. *Apache TomEE*. En: www.apache.org [en línea]. Disponible en: http://tomee.apache.org/apache-tomee.html [consulta: 31 Agosto 2015]

[pJCon4]. *Glassfish*. En: www.java.net [en línea]. Disponible en: https://glassfish.java.net [consulta: 31 Agosto 2015]

[pJCon5]. *Resin*. En: caucho.com [en línea]. Disponible en: http://caucho.com [consulta: 31 Agosto 2015]

[pJCon6]. *Wildfly*. En: wildfly.org [en línea]. Disponible en: http://wildfly.org [consulta: 31 Agosto 2015]

[pJNDI]. *Java Native Interface Specification*. En: docs.oracle.com [en línea]. Disponible en: http://docs.oracle.com/javase/7/docs/technotes/guides/jni/spec/jniTOC.html [consulta: 31 Agosto 2015]

[pJS01]. *jQuery*. En: jquery.com [en línea]. Disponible en: https://jquery.com [consulta: 31 Agosto 2015]

[pJS02]. *Dojo Toolkit*. En: dojotoolkit.org [en línea]. Disponible en: https://dojotoolkit.org [consulta: 31 Agosto 2015]

[pJS03]. *Kendo UI*. En: www.telerik.com [en línea]. Disponible en: http://www.telerik.com/kendo-ui [consulta: 31 Agosto 2015]

[pJS04]. *Ember.js*. En: emberjs.com [en línea]. Disponible en: http://emberjs.com [consulta: 31 Agosto 2015]

[pJS05]. *React.js*. En: facebook.github.io [en línea]. Disponible en: http://facebook.github.io/react [consulta: 31 Agosto 2015]

[pJS06]. *MooTools*. En: mootools.net [en línea]. Disponible en: http://mootools.net [consulta: 31 Agosto 2015]

[pJS07]. *Socket.io*. En: socket.io [en línea]. Disponible en: http://socket.io [consulta: 31 Agosto 2015]

[pJS08]. *Backbone.js*. En: backbonejs.org [en línea]. Disponible en: http://backbonejs.org [consulta: 31 Agosto 2015]

[pTWA1]. *JTwain – A Java Twain Implementation*. En: asprise.com [en línea]. Disponible en: http://asprise.com/product/jtwain/ [consulta: 31 Agosto 2015]

[pTWA2]. *Morena 7*. En: www.gnome.sk [en línea]. Disponible en: http://www.gnome.sk/Morena/morena.html [consulta: 31 Agosto 2015]

[pMSF01]. *Microsoft System Center*. En: www.microsoft.com [en línea]. Disponible en: http://www.microsoft.com/en-us/server-cloud/products/system-center-2012-r2 [consulta: 31 Agosto 2015]

[pVMW01]. *VMware vCloud Suite*. En: www.vmware.com [en línea]. Disponible en: http://www.vmware.com/products/vcloud-suite [consulta: 31 Agosto 2015]